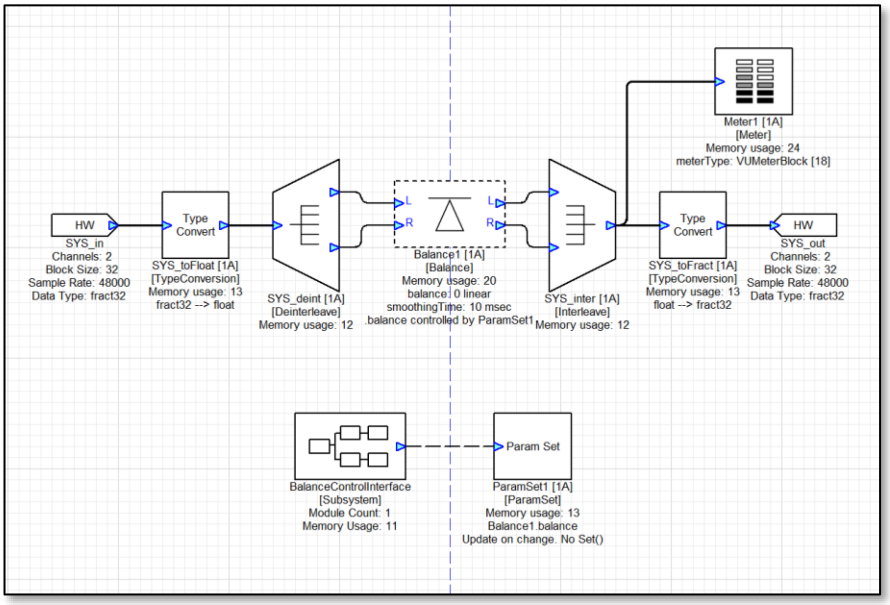
DSP Concepts

Getting and Setting Parameters and Arrays



Copyright Information

Printed in the USA.

Disclaimer

DSP Concepts, Inc. reserves the right to change this product without prior notice. Information furnished by DSP Concepts is believed to be accurate and reliable. However, no responsibility is assumed by DSP Concepts for its use; nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under the patent rights of DSP Concepts, Inc.

Change Log

Version	Date	Description
1.0	12/09/2022	Initial Draft

Table of Contents

1 About This Guide 6

2 Param Set Module 7

3 Param Get Module.... 10

4 Array Set & Param Set Table Modules.... 12

4.1 Array Set Module.... 12

4.2 Param Set Table Module 14

Appendix A: Modules Controlled by the Get/Set Modules 17

List of Figures

Figure 1: Wired Control Signal 6
Figure 2: Misc Module Folder 6
Figure 3: ParamSet Module 7
Figure 4: ParamSet View Properties 7
Figure 5: ParamSet Arguments Tab 8
Figure 6: Mod.Var Syntax........ 8
Figure 7: ParamSet setBehavior Properties 9
Figure 8: setBehavior Value Definitions........ 9
Figure 9: ParamGet Module........ 10
Figure 10: ParamGet View Properties 10
Figure 11: ParamGet Arguments Tab 10
Figure 12: ArraySet Module 12
Figure 13: ArraySet Module Properties 13
Figure 14: ArraySet Variables Properties Tab 13
Figure 15: ArraySet setBehavior Values........ 13
Figure 16: ParamSet Table Module 14
Figure 17: ParamSet Table Variables Properties 14
Figure 18: ParamSet Table Arguments Tab 15
Figure 19: ParamSet Table 'Array: coeff' Properties Tab........ 15
Figure 20: ParamSet Table Import from .csv 16
Figure 21: ParamSet Table Save to .csv 16
Figure 22: ParamSet Table Coeff Array Save/Cancel Changes 16
Figure 23: Module Controlled by Get/Set Modules 17

1 About This Guide

This Application Note describes the ParamGet, ParamSet and ArraySet modules, and their respective usage.

Wired control signals (Figure 1) such as the DC Source module, can be used to adjust parameters of modules, but this can also be accomplished with the ParamGet and ParamSet modules.

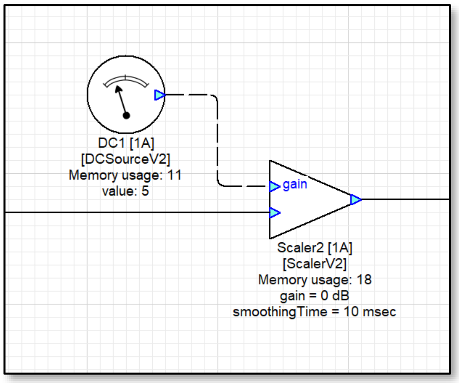


Figure 1: Wired Control Signal

The ParamGet and ParamSet modules read and write parameters to and from existing modules in an Audio Weaver Designer layout, without the need of a wired connection.

These modules can be found in the 'Misc' module folder in Audio Weaver Designer's module palette on the left-hand side of the Designer window:

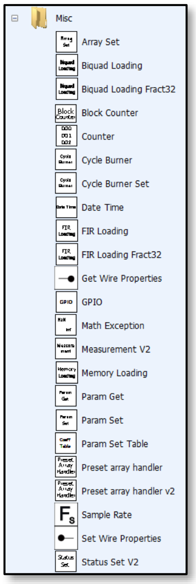


Figure 2: Misc Module Folder

2 Param Set Module

The ParamSet module takes a signal from an input wire and writes the incoming value to an instance variable in another module:

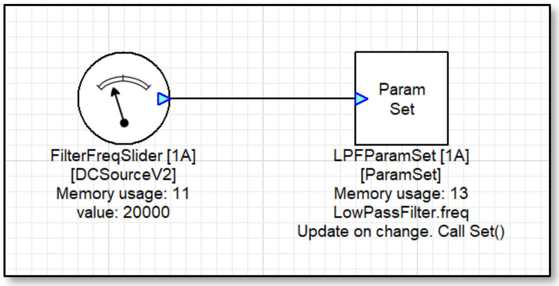


Figure 3: ParamSet Module

This module is useful for controlling the behavior of other modules in response to control signals or audio signals. This module can set parameters of any type of module, which eliminates the need to use and/or create controllable permutations of each module. The input wire can have any number of channels and any block size. The module only uses the first value in the wire and ignores all others. Internally the module uses two pointers. The first points to the base of the module being controlled. The second pointer points at the variable being written.

To set which module and variable the ParamSet module controls, right click on the ParamSet module while Audio Weaver is in Design Mode, and select 'View Properties' in the context menu:

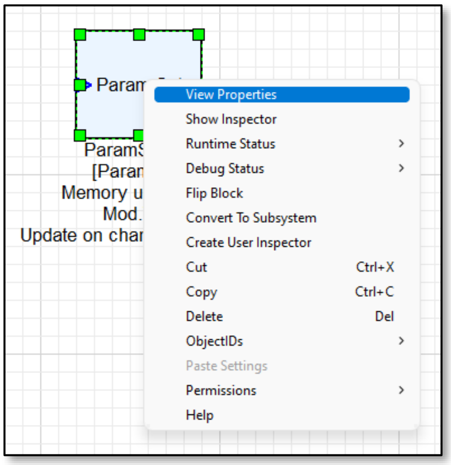
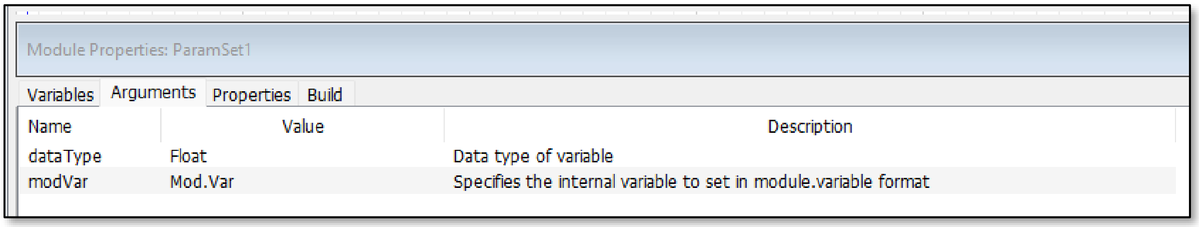


Figure 4: ParamSet View Properties

The ParamSet module's properties menu will appear at the bottom of the Audio Weaver Designer window. Navigate to the 'Arguments' tab to assign the desired module and variable you wish to control:



Name	Value	Description
dataType	Float	Data type of variable
modVar	Mod.Var	Specifies the internal variable to set in module.variable format

Figure 5: ParamSet Arguments Tab

Pictured above in Figure 5, you can see two arguments for the ParamSet module:

- `dataType`
- `modVar`

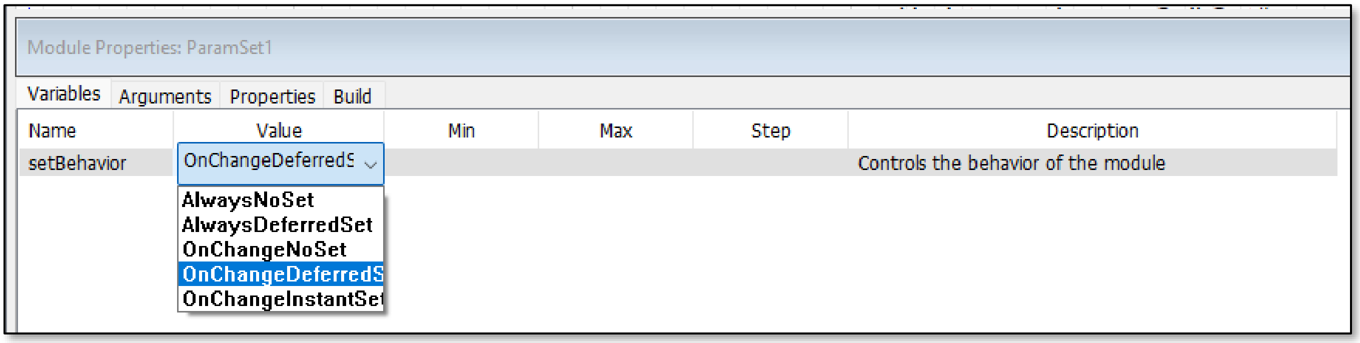
The `dataType` argument specifies the input pin's type of data (float, int, fract32).

The `modVar` argument specifies which module's variable the ParamSet module will set. The following table describes valid `modVar` syntax:

Mod.Var Syntax	Description
`Mod.Var`	Mod is the name of the module (in the same level of subsystem hierarchy as the ParamSet module) and Var is the variable name to set.
`Subsystem.Mod.Var.`	Specifies modules that are within subsystems.
`\Scaler1.gain`	To go up in subsystem hierarchy and control modules that are in an upper system, use a single backslash character to move up one level in the hierarchy. Use multiple backslash characters to traverse multiple levels of hierarchy.
`Mod.Var[index]`	Write to individual values in arrays, where index is the zero-based index of the element in the array Var to set.

Figure 6: Mod.Var Syntax

The behavior of how often the instance variable is updated and whether the module's set function is called, can be set or changed in the 'Variables' tab of the ParamSet module via the `setBehavior` variable:



Module Properties: ParamSet1

Name	Value	Min	Max	Step	Description
setBehavior	OnChangeDeferredS				Controls the behavior of the module

Figure 7: ParamSet setBehavior Properties

The `setBehavior` variable presents 5 behavioral options, but behind the scenes it is a bit field:

- Bit 0 specifies whether the instance variable is always updated (=0) or only if the value on the input wire changes (=1)
- Bit 1 specifies whether the set function of the module being control is called. (=0 do not call set. =1 call set function).

There are 5 possible bit combinations available:

setBehavior Value	Bit 0	Bit 1
0: Always update. Do not call set function (useful)	0	0
1: Always update. Call set function (not too useful)	0	1
2: Update only when input value changes. Do not call set function (useful)	1	0
3: Update only when input value changes. Call set function (Most useful)	1	1
4: Update only when input value changes. Call set function immediately (not deferred)	1	1

Figure 8: setBehavior Value Definitions

Note that in cases 1 and 3 the set function is triggered by setting the deferred bit in the module instance structure which causes the module's set function to be called as a background task. It may take tens of milliseconds before the set function is actually called and the exact timing can depend upon the CPU loading of the system. If the system is lightly loaded then there is more CPU power available for performing the deferred processing tasks. For case 4, the set function is called immediately in the real-time thread. Use with caution since this increases the CPU load.

An example layout ('ParamSet_Example.awd') further explaining the ParamSet module usage can be found in the 'C:\DSP Concepts\<AWE Designer>\Examples\Module_Usage' folder of the default Audio Weaver Designer installation path.

3 Param Get Module

Similar to the ParamSet module but opposite in functionality, the ParamGet module reads the value of a parameter from an existing module and outputs it on a wire for use elsewhere in the system.

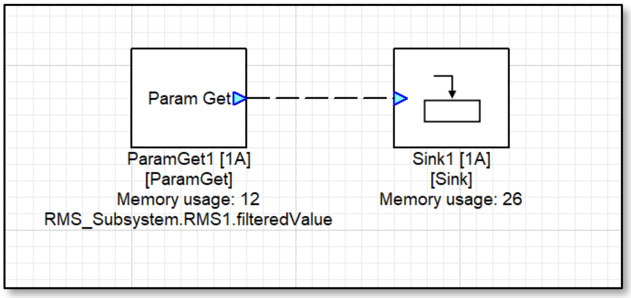


Figure 9: ParamGet Module

Internally the module uses two pointers. The first points to the base of the module to get. The second pointer points at the variable to get within the module instance structure.

The parameter which the ParamGet module outputs is specified in the module properties. To set which module and variable the ParamGet module controls, right click on the ParamGet module while Audio Weaver is in Design Mode, and select 'View Properties' in the context menu:

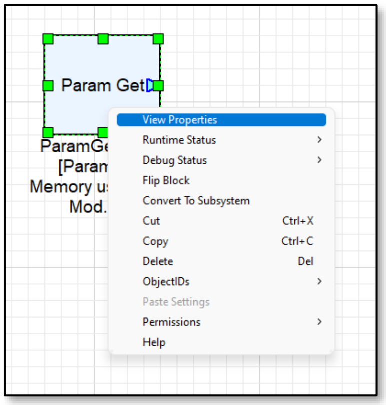
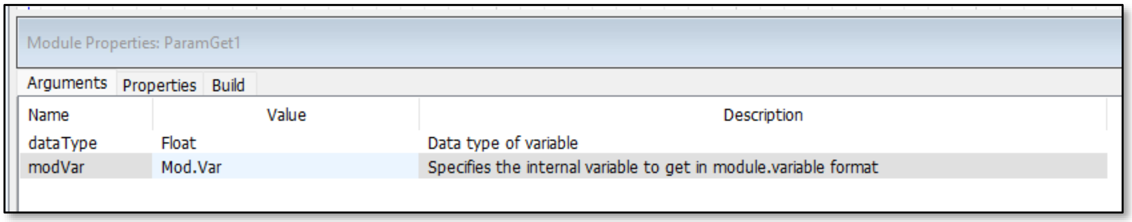


Figure 10: ParamGet View Properties

The ParamGet module's properties menu will appear at the bottom of the Audio Weaver Designer window. Navigate to the 'Arguments' tab to assign the desired module and variable you wish to control:



Module Properties: ParamGet1

Arguments | Properties | Build

Name	Value	Description
dataType	Float	Data type of variable
modVar	Mod.Var	Specifies the internal variable to get in module.variable format

Figure 11: ParamGet Arguments Tab

Pictured above in Figure 11, you can see two arguments for the ParamGet module:

- `dataType`
- `modVar`

The functionality and syntax for each variable are identical to the ParamSet module, which are described in detail above on page 8.

An example layout (‘ParamGet_Example.awd’) further explaining the ParamSet module usage can be found in the ‘C:\DSP Concepts\<AWE Designer>\Examples\Module_Usage’ folder of the default Audio Weaver Designer installation path.

4 Array Set & Param Set Table Modules

Similar to the ParamSet and ParamGet modules which act on scalar values, Audio Weaver also features modules for setting array values. At the time of this application note writing, Audio Weaver does not feature an ArrayGet module. Typically, if a user needs to know the values of an array, outputting the array to a Sink Module or a .csv satisfies any ArrayGet use cases.

4.1 Array Set Module

The ArraySet module can set array values of another module, with data from input wire.

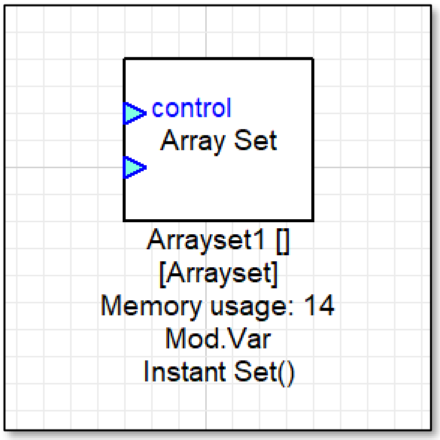
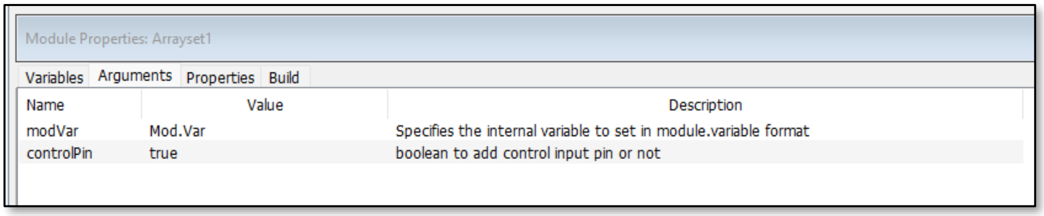


Figure 12: ArraySet Module

The ArraySet module reads the data coming in from the input wire and reaches into instance structure of the target module to copy the data to the target array. The input array size must be less than or equal to the size of the target variable array size, and the input array updates the target variable starting at the first array element. If the input pin array size is less than the target array size only the number of elements from the input array are changed in the target array.

The array data structure of the target module is pointed to by the argument `modVar`. The `modVar` value for the ArraySet module is accessed via the 'Argument' tab of the module's property window, which can be accessed by right clicking on the module and selecting 'View Properties' in the context menu, like any Audio Weaver module.

The syntax for the ArraySet module's `modVar` argument follows the ParamSet and ParamGet `modVar` syntax outlined above on page 8, in Figure 6.

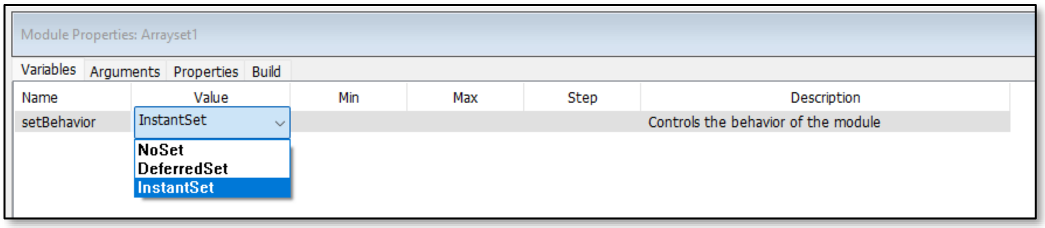


Name	Value	Description
modVar	Mod.Var	Specifies the internal variable to set in module.variable format
controlPin	true	boolean to add control input pin or not

Figure 13: ArraySet Module Properties

Unlike the ParamSet and ParamGet modules, the ArraySet module has an optional control pin that can be enabled or disabled in the 'Arguments' tab of the module's properties. The control pin enables copying the input data to the `modVar` array when the first value of the control pin data !=0, acting as a type of trigger. If the control pin is not used, then the data is written to the target array on every execution.

Similar to the ParamSet and ParamGet modules, the ArraySet module's 'Variables' tab in the Properties menu, contains an option to set the behavior of the module via the `setBehavior` variable:



Name	Value	Min	Max	Step	Description
setBehavior	InstantSet				Controls the behavior of the module

Figure 14: ArraySet Variables Properties Tab

Below is a table describing each of the ArraySet module's `setBehavior` values.

setBehavior Value	Description
NoSet	Updates target module array, but never calls the target module's set function
DeferredSet	Updates target module array, and uses a deferred call to the target module's set function so it executes after processing is complete
InstantSet	Updates target module array, and immediately calls the target module's set function during the Array Set process

Figure 15: ArraySet setBehavior Values

4.2 Param Set Table Module

The ParamSet Table module is similar to the ArraySet module but offers slightly different features for different use cases. The ParamSet Table module writes the values of one column of a statically defined table to an internal array of another module. A common use case is to update the coefficients of an FIR module based on changing conditions in the system.

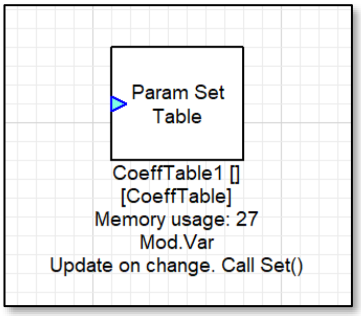
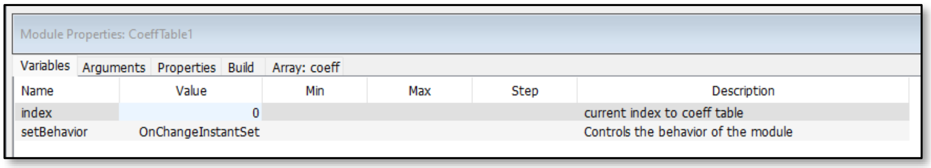


Figure 16: ParamSet Table Module

Let's take a look at the ParamSet Table Module's Properties. In the 'Variables' tab of the module's Properties, we have two variables, `index` and `setBehavior`:



Name	Value	Min	Max	Step	Description
index	0				current index to coeff table
setBehavior	OnChangeInstantSet				Controls the behavior of the module

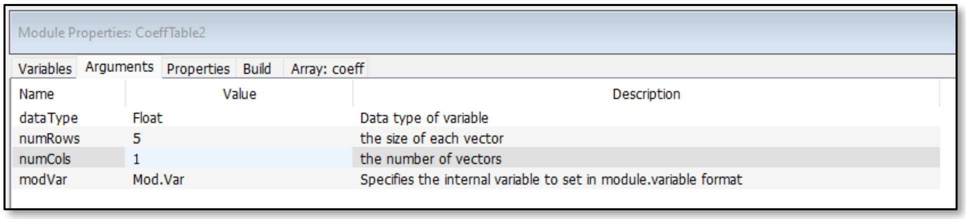
Figure 17: ParamSet Table Variables Properties

The `setBehavior` values and resultant behavior are identical to the ParamSet module's, described above on page 9, in Figure 8.

The `index` variable specifies the current coeff column index of the coeff table defined in the 'Array: coeff' tab, to select which column to copy to the target module's array. This variable can be changed here in the 'Variables' tab of the module's Properties and also by the integer value of the input pin.

In the 'Arguments' tab, we can see the following fields:

- `dataType`
- `numRows`
- `numCols`
- `modVar`



Name	Value	Description
dataType	Float	Data type of variable
numRows	5	the size of each vector
numCols	1	the number of vectors
modVar	Mod.Var	Specifies the internal variable to set in module.variable format

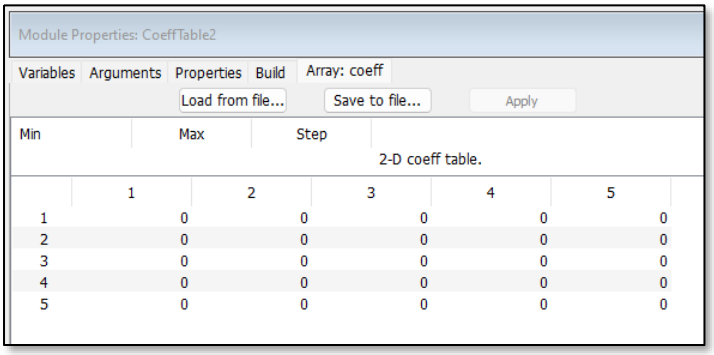
Figure 18: ParamSet Table Arguments Tab

The dataType argument specifies the data type of the values in the coeffs array, defined in the module's 'Array:coeff' Properties tab (float, int, fract32).

The numRows variable determines the length of each vector, and the numCols variable determines the number of vectors that can be stored in the internal table. It is the responsibility of the user to ensure that the numRows variable matches the size of the target module's array. If numRows and numCols are all = 1 then the module will assume a scalar value to write.

The modVar argument specifies the target module and array variable that the ParamSet Table module controls. The syntax is identical to the ParamSet syntax defined above on page 8, in Figure 6.

Skipping over the 'Properties' and 'Build' tabs, we reach the 'Array: coeff' tab of the ParamSet Table's Properties menu:



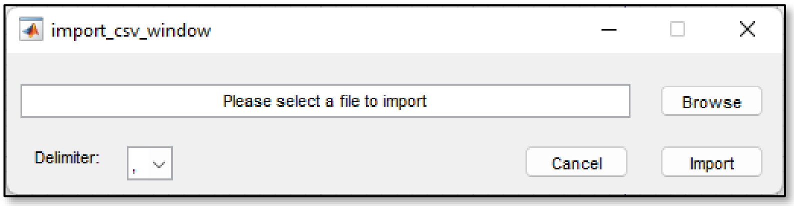
2-D coeff table.

	1	2	3	4	5
1	0	0	0	0	0
2	0	0	0	0	0
3	0	0	0	0	0
4	0	0	0	0	0
5	0	0	0	0	0

Figure 19: ParamSet Table 'Array: coeff' Properties Tab

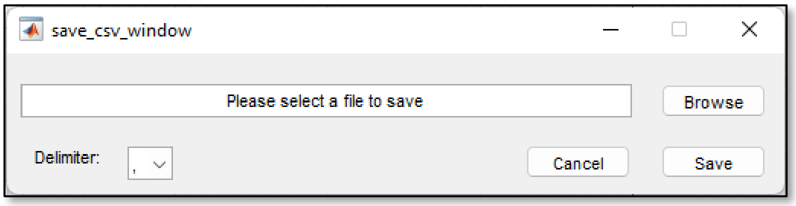
As described above, this table's dimensions are specified by the numCols and numRows variables in the 'Arguments' tab, and the column holding the array that gets written to the target modVar (specified in the 'Arguments' tab) gets set by the ParamSet Table's input pin AND/OR the index variable specified in the 'Variables' tab.

The ParamSet Table module features options to either load a coeff table from a .csv file or save the current table to a .csv file:



Please select a file to import

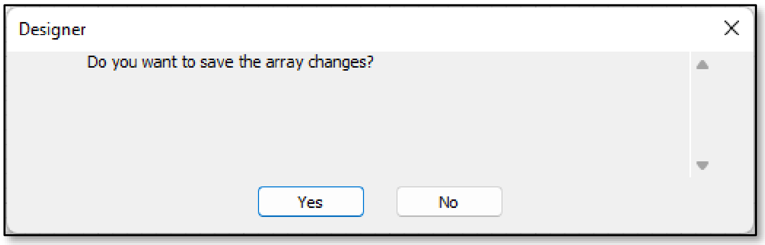
Figure 20: ParamSet Table Import from .csv



Please select a file to save

Figure 21: ParamSet Table Save to .csv

Lastly, any time a change is made to the coeff table in the 'Array: coeff' tab, Audio Weaver Designer will prompt to either save or discard the changes made to the array:

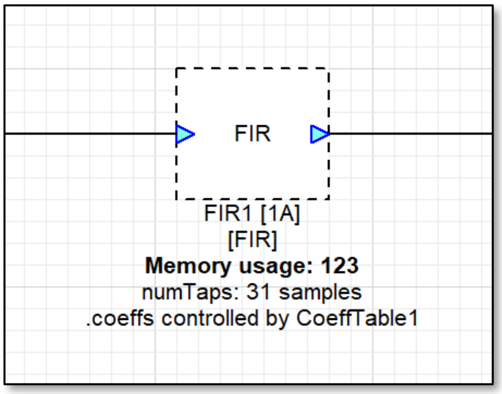


Do you want to save the array changes?

Figure 22: ParamSet Table Coeff Array Save/Cancel Changes

Appendix A: Modules Controlled by the Get/Set Modules

Any module controlled by one of the get/set modules described in this application note, will visually change and display which module it is controlled by in the Audio Weaver Designer canvas, in text under the module. As seen below in Figure 23, the FIR1 module has a dashed outline instead of a solid border, and the text under the module indicate that it's coeffs array variable is controlled by the CoeffTable1 module:



FIR1 [1A]
[FIR]
Memory usage: 123
numTaps: 31 samples
.coeffs controlled by CoeffTable1

Figure 23: Module Controlled by Get/Set Modules