LJ SCS Content Expander v002-001.html documentation

Overview

This JavaScript / HTML file enables you to create a TAB delimited file from Excel that contains information to manage x32 Channel Strip content, DCA Membership, load and save snippets, Automix on/off and assignments, Bus Sends on and off settings and Mute/Unmute status for channels, auxs, buss, etc. in a fairly easy to use manner by augmenting command building capabilities of Show Cue Systems (SCS) (https://www.showcuesystems.com/cms/). You can also leverage the power of Excel and vlookup and other functions to allow for easy manipulation of cue file content. This solution also allows you to manage cues for multiple x32s concurrently.

Why This software?

Show Cue Systems (SCS) is a powerful tool for managing audio cues, light cues, image displays, video displays, etc. as well as for controlling external devices or triggering external light or sound cues and it has a built in OSC interface to Behringer x32s. However, the management of large numbers of channels, DCAs, Matrixes, etc. using the data entry and editing interface in SCS is tedious and difficult to visualize. This is especially problematic when there are many dynamic audio requirements, such as when running theater production via DCA groups or muting and unmuting many channels, Matrixes, AUXs, etc.

This simple SCS Expander script lets you create your complex sound cue content in Excel, even leveraging the power of Excel functions and lookup capabilities. Then, in SCS, you create a cue structure into which the complex content in Excel is to be placed. This expander software reads the SCS file, looks for placeholder cue information you placed there and expands it with information from the Excel file (via a tab delimited save of the excel file) and provides you with the resultant content to use as a complete expanded SCS file.

You can also place saves and loads of Snippets in cues. This can be very handy in mic swap situations where you can define a snippet to save the channel attributes you want and then have them saved in the run of your show after setting up a user and before placing another user on that same mic. This lets you keep the latest and greatest settings without having to separately manually save a snippet while running the show. You can also recall it at the next run of the show or whenever that mic user re-takes that mic. This is a great way to keep the latest saved with your scene.

NOTE: Although the software was tested, it is up to you to review all generated content for functionality and completeness prior to using it. The software author is not liable for any use made of this software. All generated content should be tested prior to use in a critical situation.

Components Tested:

SCS: This version was tested with SCS Professional Plus version 11.8.2.1 . *The version of SCS licensed needs to support the number of cues and use of a Control Send device. The SCS software is licensed separately.*

Windows: Windows 10, 64 bit was used for testing.

Browser: Google Chrome 80.0.3987.112 64 bas used for testing.

DETAILS

In SCS, create your general Cues as normal, but also build in placeholders for the sound cues that you want to establish through this SCS Expander software.

Build the bulk of the cue content in Excel, save as Tab Delimited, run the SCS Expander software, specify the SCS base and substitution tab delimited Rules files, and you get the new SCS file content with the Excel-defined content merged in following the associated SCS Cue placeholders.

Here's an example:

Open SCS and create a new or open an old Production. Be sure your network connects to your x32 so you can send commands.

With SCS 11.8.2.1 you can use the Editor to get to Production->Properties. Then under Devices go to Control Send and set up a connection to your x32. *(Note that if you have trouble getting the port set, change the device to an x32 compact…which will enter the port value…then change pack to an x32 and enter the IP. Seems there's a bug in this version of the SCS software*.) Be sure to note what all you name your x32(s) in the device map. This same name is used in your Cues, allowing the system to control multiple x32s by name from a single place.

When building this base SCS cue file, in the Cue positions where you will want to get commands from the expansion from the Excel content, create a control send cue as a placeholder. In the Description show what you would like…maybe something like "Page 27 Cue 01" to show this cue is to be run on page 27 of the script as the first cue on that page….or however you like to designate these things. Name the Cue what you want, place the "when required" text as you would like.

In the Control Send Cue for the OSC Command Type select Free Format. Then for the OSC Message place a unique string so that this SCS Expander software can find that placeholder string and insert the desired commands after it in the SCS XML file. For example, you could use:

/|Cue027-01|

(the leading slash makes SCS accept the syntax of this content).

That's it. Do this for as many placeholder cues you want expanded by the SCS Expander software. If you want, you can even create cue content in the Excel file that you may want to repeat in the SCS cue list…maybe to set a base chunk of values you will often use to reset things. This is fine, just create that content once in the Excel file and duplicate the Placeholder's OSC Message multiple time in the Base SCS file. It is the OSC Message that acts as the placeholder for content from the SCS Expander software.

SCS will save this within the content of its XML file you just saved, as in this example where a free format cue is specified as /|*CUE027-01*|. The device name for this x32 was set as "FOH X32".

```
<Cue>
    <CueId>SETUP</CueId>
    <Description>pg 27 setup - unmute CH1-3 and place in DCAs</Description>
```

```
    <PageNo>27</PageNo>
    <WhenReqd>Sets starting conditions</WhenReqd>
    <Sub>
      <SubType>M</SubType>
      <SubDescription>X32 OSC</SubDescription>
      <DefSubDes>1</DefSubDes>
      <ControlMessage>
        <CMLogicalDev>FOH X32</CMLogicalDev>
        <OSCCmdType>free</OSCCmdType>
        <OSCItemString>/|Cue027-01|</OSCItemString>
        <OSCData>/|Cue027-01|</OSCData>
      </ControlMessage>
    </Sub>
  </Cue>
```

See the reference to this cue you just created? The SCS Expander Software will be used to read the XML file from SCS and rewrite it while inserting audio cues you define in Excel. It will leverage the specified tag in the Excel generated tab delimited file to determine where to do expansion (in this case /|Cue027-01|).

Here's how the Base SCS information appears in the SCS software.

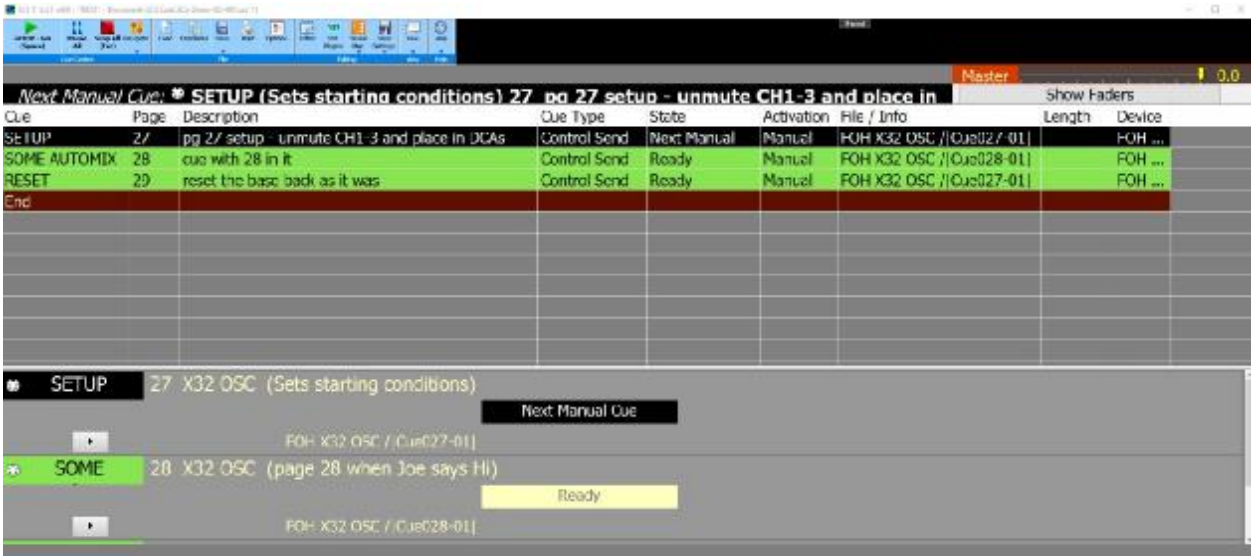

Figure 1 Base SCS Content Main Window

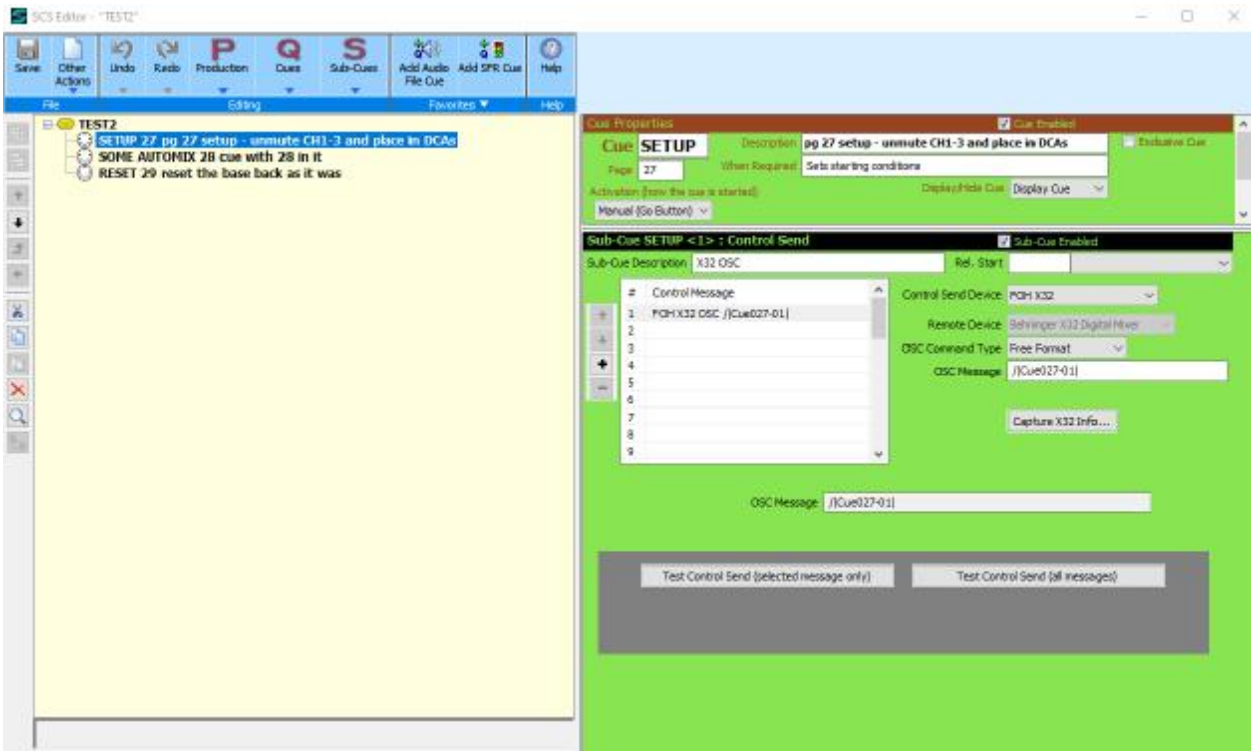

Figure 2 Base SCS Content Edit Window

Specific details on the available Excel content will be provided later, but for now, let's consider an example to provide the concept. Suppose we want to color several channel strips White to indicate that they are important and we want to place several channels into one or more DCAs and unmute the DCAs and the Channels and name the DCAs something and enable some Sends for the channels and load and then save a Snippet. For this example, the Channels already have great names on their scribble strips

and we are not changing them. Then later you want to add Channel 1 to Automix X and Channel 2 to Automix Y and turn Automix X off and Y on and color the involved channels Red

We would create an Excel file like this:

	A	B	C	D	E	F	G	H
1	~comment	Demo Sample						
2	~comment	The HEADER line defines the subsequent column content						
3	~DEV	X32	FOH X32					
4	~TAGSTART	|Cue027-01|	~	use for page 27 setup				
5	~sHEADER	ch-01	ch-02	ch-03	dca-1	dca-2	dca-3	~
6	~sNAME	^	^	^	Joe	Sam	All	~
7	~sCOLOR	wh	wh	wh	wh	wh	wh	~
8	~sLOADSNIPPET	047	~					
9	~sON	1	1	1	1	1	1	~
10	~sSAVESNIPPET	47	SuperSnip	~				
11	~sDCA	1 3	1 3	^	^	^	^	~
12	~sSENDon	01 03 05	02 04 06	01 02 03	~			
13	~TAGEND							
14	~TAGSTART	|Cue028-01|	~	place some automix and color them				
15	~sHEADER	ch-01	ch-02	ch-03	~			
16	~sAUTOMIX	x	y	off	~			
17	~sAUTOMIXx	0	~					
18	~sAUTOMIXy	1	~					
19	~sCOLOR	re	re	re	~			
20	~sSENDoff	01 03 05	02 04 06	01 02 03	~			
21	~TAGEND							
22	~DONE							
23								
24								

Figure 3 Excel Tab file content example

Here is what this is saying.

Any row beginning with ~comment is a comment and the tool will ignore it.

The ~s at the start of commands is a naming convention that indicates these are commands used for sound control.

The ~DEV command indicates that all items until the next ~DEV command apply to an X32 named FOH X32. This allows you to write scripts that control multiple x32 devices by name.

The rows starting with ~TAGSTART indicates the Placeholder tag name to look for in the SCS file (note the leading slash in the name is omitted here).

The ~sHEADER rows identifies the x32 components to work on for what will be present in the rows below in the same column (every ~sHEADER row replaces any prior ~sHEADER row.)

The ~sNAME row indicates what to place in the scribble strip on the x32 for the item in the associated ~HEADER column. A carat symbol (^) indicates that this entry should be ignored. Some names are already set so no need to modify them. In this example, we will provide names for DCA 1, DCA 2 and DCA 3.

The ~sCOLOR row says what color to set the scribble strip. We are making all of them white.

The ~sLOADSNIPPET loads the console's Snippet number 047. Keep in mind that the contents of the snippet are set when the snippet is established.

The ~sON row indicates unmuted (1) or muted (0) and we are turning them all on (unmuted).

The ~sSAVESNIPPET loads the console's Snippet number 047 and sets the name for that Snippet. It does NOT change the list of attributes associated with that Snippet.. Keep in mind that the contents of the snippet are set when the snippet is established.

The ~sDCA row indicates which DCA(s) an entry should be in. Of course a DCA cannot be in a DCA so we use the ^ to ignore that as well. We are placing Channels 1 and 2 into DCAs 1 and 3.

The ~sSENDon row indicates which that we want Channel 1 Bus Sends 1, 3, and 5 on; Channel 2 on for Bus Sends 2, 4 and 6; and Channel 3 Bus Sends 1, 2 and 3 on.

The ~TAGEND indicates that this is everything we want done for this tag's position in the SCS file.

We add another tag for another Placeholder in the base SCS file.

In the second Tag we place more whatever commands we will want expanded into the SCS file when this Tag is found.

The ~sAUTOMIX row places Channel 1 in Automix x and Channel 2 in Automix y and remove Channel 3 form Automix.

The ~sAUTOMIXx row does not depend on a column, it is just a config setting, so it just turns off Automix X.

The ~sAUTOMIXy row is similar, but turns Automix Y on.

We then color these 3 channels Red.

The ~sSENDoff row turns off the sends we turned on in the |Cue027-01| cue.

We then end this tag.

The ~DONE row indicates that we have completed the entire rules content definition for all tags we want to have processed.

We then save this as a Tab Delimited file in Excel. It is saved as a .txt file with tab delimiters.

Next we run the tool and tell it to open the SCS file that the SCS software created.

Then tell it to open the SCS RULES file (which is the tab delimited file you just saved from Excel).

Then a text area of the expanded SCS content is displayed for you to copy and paste into your new and expanded SCS file.

Below is an illustration of what the SCS Expander script presents to the user. In this example we are using Google Chrome as the browser that opened the tool. In this illustration, the top green area displays the base SCS file content that you selected. The yellow section displays the tab delimited Rules file you selected. The white section displays log info in case something goes wrong. The dark blue area with white text contains the merged content result from expanding the base file using the rules you provided.

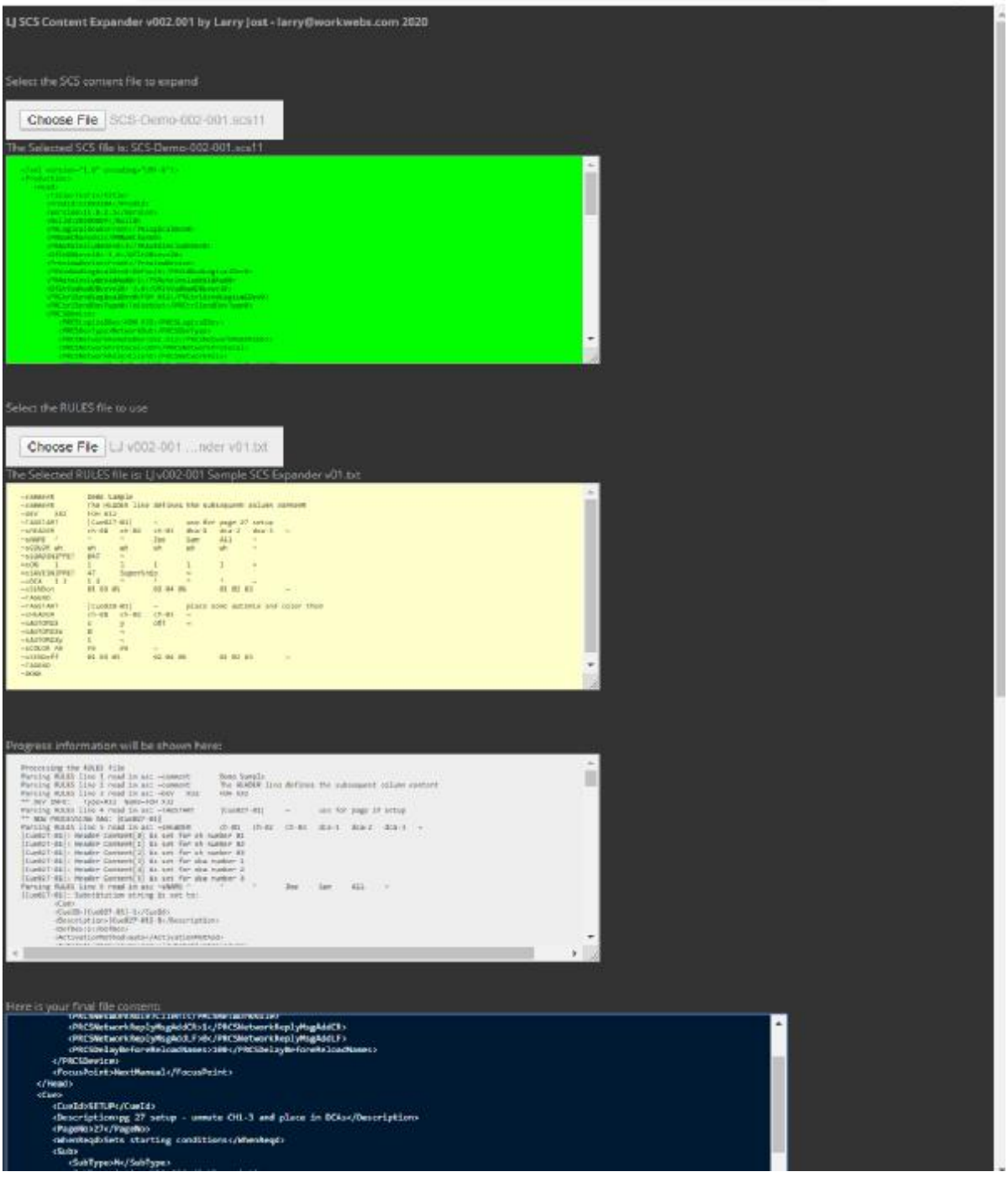

Figure 4 SCS Expander browser window

Select All of the content in that window (in Windows Control+a) and copy it to the clipboard (in Windows Control+c). Save that into a .scsxx file (xx is the SCS version number as mimicked from the scs source file extension) in the folder where your other SCS files are (default is Documents -> SCS Cues). It is suggested that you keep the same name as the original file but add the word "Expanded" to it so that you know it is the expanded version. In Windows, it is recommended that you paste it into a new Notepad file (in Windows Control+v) to avoid adding any control characters to the file.

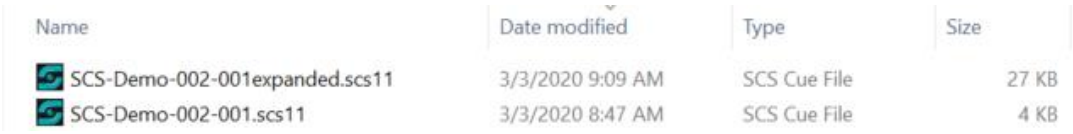

Figure 5 Sample naming of files

Start SCS and browse to the expanded file and open it. It does not look like much has happened, but there are now hidden cues under each of these cues that will auto fire after the associated visible cues to execute the expanded functions that are now in place.

Figure 6 Main SCS window after expansion file read in

Use the SCS Editor to check content or even edit it. Note that the visible cue is also run, but it contains an invalid command for the x32 so is ignored. Be careful that your Placeholder OSC Command tags are not going to cause problems. The next image shows how you will see the cue lists generated for you.

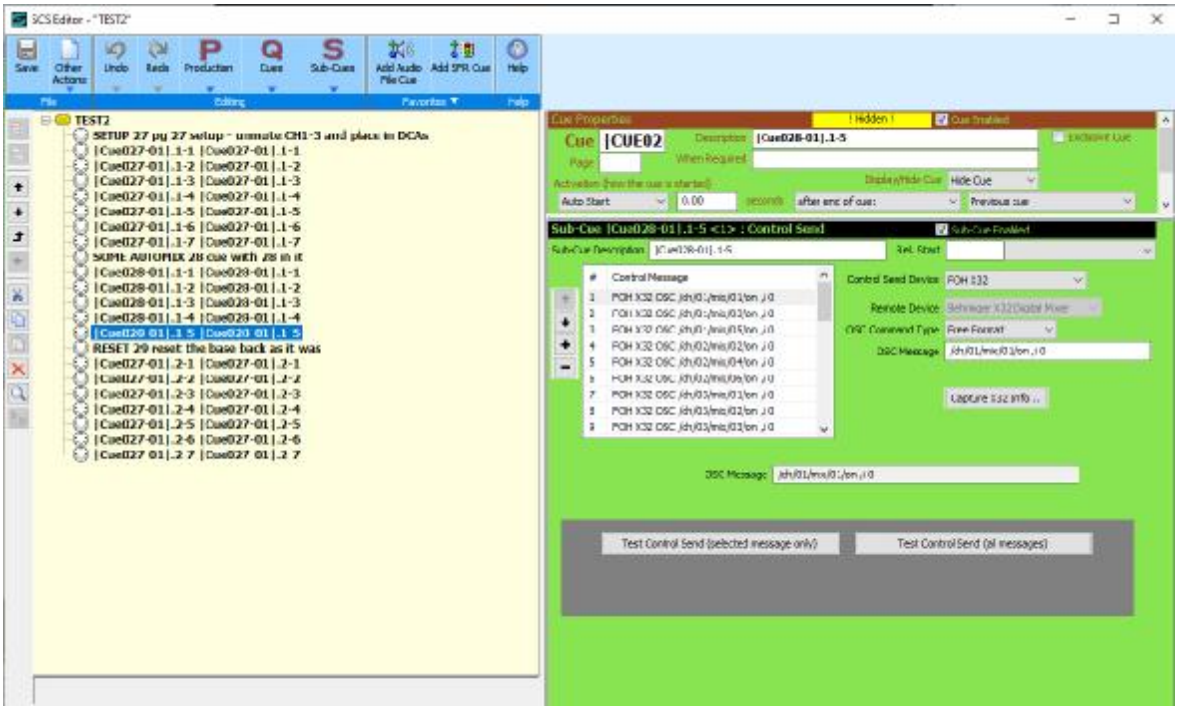

Figure 7 SCS Edit window after opening the expanded SCS file

Within each of these cue parts you will see the details of the expansion. The naming adds a repeat count for each use of a placeholder as well as a part number form the cue expansion. SCS only allows 32 entries in each cue, but this software is set to use less so that you can augment these later without much work later on.

NOTES:

KEEPING CURRENT CHANGES WITH SNIPPET SAVING IN THE CUES

If you pre-establish a Snippet for each user of a mic on a set channel and save these on the X32 and mark them for what you would like saved, then you can use cues to load this Snippet prior to each use of a mic by the person it is saved for. You can also save the Snippet before replacing it so that your latest changes from a run of your production are saved in that snippet for the next time that person uses that mic. The end of show cue could save them all so they are all up to date with the latest prior to the next run. If you do this, also save Scenes via the X32 before the show so you can revert to the pre-show state as well.

DYNAMIC INFORMATION VIA EXCEL

Keep in mind that Excel formulas, lookups, etc. can be used. Using a COMMENT in a row that is followed by as row of data you would like referenced from a lookup table allows you to use vlookup or other functions to make maintenance of the cues easier.

SCS AND THE XML

Keep in mind that the your base starting point for the Cue definitions is an important asset to maintain. Once you do an expansion and move forward any changes you make in the expanded XML are not in the original base. Be careful to manage changes in a manner that you can maintain them with. If you transition to editing the expanded file, then you cannot go back to the base and regenerate things without losing those changes unless oyu also update the Excel file to re-generate the XML with those changes in place. Just keep that in mind.

COMMAND DETAILS

Here are the commands that are allowed in the first column of the RULES Tab delimited file along with overall information.

Any double-quote character in the Tab Delimited file will be removed.

Every ~DEV entry replaces any prior ~DEV entry so you can change these any time you want. However, it is up to you to be sure that there is always a ~DEV entry in place prior to any command that acts on a Device.

Every ~sHEADER entry replaces any prior ~sHEADER entry so you can change these any time you want. However, it is up to you to be sure that there is always a ~sHEADER entry in place prior to any command that requires the header content.

Note that the last meaningful column used in any row that has more than one column of potential content must be followed by a ~ character. Anything after that in this same row is treated as a comment.

As a naming convention, commands that begin with ~s are for sound cues.

~comment

Any line in the rules file that begins with ~comment is ignored later on.

Note that the last meaningful column used in any row that has more than one column of potential content must be followed by a ~ character. Anything after that in this same row is treated as a comment.

~DEV *with the next column containing a Device Type and the next column a Device Name*

There is presently only one implemented value for the Device Type specification and that is X32

The Device Name must match the device name used in the SCS Control Device "Name Used in Cues". This is because SCS will send the command to the device based on this value.

You may have multiple ~DEV commands in the rules file. Each one replaces any prior ones.

A ~DEV command must be in place prior to any command that will send a console command.

~TAGSTART *with the next column containing a tag*

The Tag specified is what you use in the OSC command in the Cue in SCS for which subsequent rows up to the next ~ENDTAG will be used for added information to place into the SCS cue set. Be sure to make these something that will not occur elsewhere in the SCS file. Here, the tag does not include a leading / like it does when you add it to the SCS free format command.

You may place as many ~TAGSTART … content … ~TAGEND sets of rows in the file as you want. DO NOT repeat any Tag names in the Excel file or strange things may happen when it is used. Note that you can repeat use of these Tags in the original base SCS file so you get the same expansion each time the tag is used there.

~TAGEND

This command indicates that the substitution content for the prior ~TAGSTART has ended.

~DONE

This command indicates that the end of the content has been reached.

At present, the Following Commands have only been implemented for Device Type X32.

~sHEADER *with subsequent columns specifying a channel, bus, auxin, fx return, matrix, or bus*

The ~sHEADER defines which item each following row in that same column applies to. You can identify a channel, aux-in, bus, fx-rtn, DCA, or Matrix.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

You may place new ~sHEADER rows whenever you want in the Excel file to establish new designations. They linger until a new one is read in. Be sure to have one of these prior to any command rows that require header content or things will not work right.

There are two formats used for this command.

The first format is:

x-yy

where x is

ch for a channel designation

auxin for an aux-in designation

bus for a bus designation

fxrtn for a function return designation

where yy is a two digit number. For 0 through 9 use 00 through 09

The second format is:

x-y

where x is

dca for a DCA designation

mtx for an mtx designation

where y is a single digit number

~sNAME *with subsequent columns specifying scribble strip content for the entity identified in the prior header row.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

~sCOLOR *with subsequent columns specifying a color value for the scribble strip of the entity identified in the prior header row.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

The following are valid colors:

of for off
re for red
gr for green
ye for yellow
bl for blue
ma for magenta
cy for cyan
wh for white
ofi for off inverted

rei for red inverted
gri for green inverted
yei for yellow inverted
bli for blue inverted
mai for magenta inverted
cyi for cyan inverted
whi for white inverted

~sON *with subsequent columns specifying the off (muted) vs on (unmuted) status of the entity identified in the prior header row.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

A 1 means the entity is to be set to on (unmuted)

A 0 means the entity is to be set to off (muted)

~sDCA *with subsequent columns specifying the DCA group(s) of the entity identified in the prior header row.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

To place an entity into one or multiple DCA groups, just enter the DCA numbers desired separated by spaces in the appropriate column. Enter only a 0 to remove the entity from all DCAs.

~sAUTOMIX *with subsequent columns specifying the automix of the entity identified in the prior header row. This only works when the header row for the column specified is for channels 1-8.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

An x indicates placement into Automix X.

A y indicates placement into Automix Y.

~sAUTOMIXx *with a 0 or a 1 in the next column sets AUTOMIX X to on or off*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

A 0 turns Automix X off.

A 1 turns Automix X on.

~sAUTOMIXy *with a 0 or a 1 in the next column sets AUTOMIX Y to on or off*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

A 0 turns Automix Y off.

A 1 turns Automix Y on.

~sSENDon *with subsequent columns specifying space separated lists of Busses that should be On for the channel associated with the column as established by a ~sHEADER command.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

The format for the Bus numbers is a 2 digit integer, so for 1 through 9 use 01 through 09. If you only want to turn on 1 Send, just place one value in the cell. If you want to impact multiple Sends, just separate the Bus numbers with a space within the same cell.

~sSENDoff *with subsequent columns specifying space separated lists of Busses that should be Off for the channel associated with the column as established by a ~sHEADER command.*

A ^ in a column indicates to not change the value for the entity identified in the prior header.

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

The format for the Bus numbers is a 2 digit integer, so for 1 through 9 use 01 through 09. If you only want to turn off 1 Send, just place one value in the cell. If you want to impact multiple Sends, just separate the Bus numbers with a space within the same cell.

~sLOADSNIPPET *with a three-digit Snippet index number in the next column*

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

Note that leading 0s in Snippet index numbers appear optional, but using a 3 digit value is recommended.

~sSAVESNIPPET *with a three-digit Snippet index number in the next column and the Snippet Name to use in the next column*

The last meaningful column used must be followed by a ~ character. Anything after that in this same row is treated as a comment.

Note that leading 0s in Snippet index numbers appear optional, but using a 3 digit value is recommended.