



Using Custom Lines with Serial Devices

Broadcast Electronics Tech Note

THE INFORMATION IN THIS ARTICLE APPLIES TO:

All AudioVAULT products

SUMMARY

Custom Lines allow for broader control of serial driven devices like switchers and receivers. This helps overcome the limitations of having only 15 Indicators and 15 Macros per screen instance. This example relates to switchers specifically, but may be applied to any serial device that the AV needs to send commands to. Please note this document assumes that the computer's COMM Port is already configured and the switcher is on-line and responding.

MORE INFORMATION

The AudioVAULT software is capable of sending serial strings to control various serial devices. The most common use is for switchers. In this example, our customer is using one AVNET screen to control a BCTOOLS 12x4 Stereo switcher. Outputs 1,2 go to two remote AVNet screens located on a PC in the Production Room. Output three goes to the local PC in the On-Air Studio. This means facilities for at least 36 commands are needed, which exceeds our limit of 30 by combining indicators and macros. You may be asking yourself, why only 30? We are running 3 screens that would give a possible 90 commands? Here's why. The switcher has only one COMM Port for incoming commands. Yet we have two computers that need to talk to it. Why not use some kind of splitter? It is possible to use a Serial Splitter and give both PCs control, but this would be less desirable because of connection issues and possible collisions. Since you "need" this content to air, and want to minimize risks, the safest method to control the outputs is to have the ON-AIR PC manage switches.

What is a custom line?

In a technical definition, a custom line allows the user to define a minimum serial instruction in the `IDD_CustomX` line and then insert a variable into a playlist via the `CustomX="something"` line.

In human speak, the custom line allows you to insert custom information into a predefined command to make the switcher switch. Another way to think of it is a fill-in the blank statement. And, your `CustomX="something"` fills the blank.

There are a total of eight custom lines available. Custom1 to Custom3 are reserved for screen display.

Here's the scenario again:

Our customer is using one AVNET screen to control a BCTOOLS 12x4 Stereo switcher. Outputs 1,2 go to two remote AVNet screens located on a PC in the Production Room. Output three goes to the local PC in the On-Air Studio.

Dissecting the serial command line required to switch the switcher.

For the switcher to switch it needs the following information:

- UnitID Number
- Input Channel Name
- Output Channel Name

For the AudioVAULT to switch the switcher it needs the following information:

- Which COMM Port to use
- UnitID Number to address
- Input Channel Name
- Output Channel Name

That command line looks like this: COM1:*0013\x0D

This statement says

COM1:	Send from COM1 the following string:
*	Prefaces the Unit ID
0	Is the Unit ID
01	Is the input channel (01-12 are possible)
3	Is the output channel
\x0D	Hexadecimal code for <CR>, or Enter (required)

Or, "Use Com1 to get unit 0 to select input 1 and feed it to output 3."

A remote control section in the Audiovau.ini might look like this:

```
[BCTOOLS12X4X1]
IDD_Indicator1=COM1:*0011\x0D;NUL:25
IDD_Indicator2=COM1:*0021\x0D;NUL:26
IDD_Indicator3=COM1:*0031\x0D;NUL:27
IDD_Indicator4=COM1:*0041\x0D;NUL:28
...
```

Dissecting the custom line required to switch the switcher.

Our example uses Custom5. The line we need to use to switch this switcher is as follows. This line must be added to the AVNET screen section in our audiovau.ini to work. The lighter text is provided for example purposes and is not a complete section.

```
[AVNET]
Include=Buttons.Record3:1,BCTOOLS12x4x3
Dialog=AVNET
...
PreviewAdvance=Off
RefreshDelay1=300000
AVNET_Load=SelectDeck8
```

Deck8=ND-%a

IDD_Custom5=COM1:+*0;COM1:-\x0D;

IDD_Custom5=	When Custom5 is called,
COM1:	Send from COM1 the following string:
+*0	Takes the contents of Custom5="013" in the playlist and appends or adds *0 to the front of the line. Example: 0013 . <i>Note: *0 represents the static information to be sent every time the custom is called.</i>
;	Clears the PC's serial buffer for the next command
COM1:-\x0D	Sends hexadecimal code for <CR>, or Enter to tell the switcher that is the end off the command line

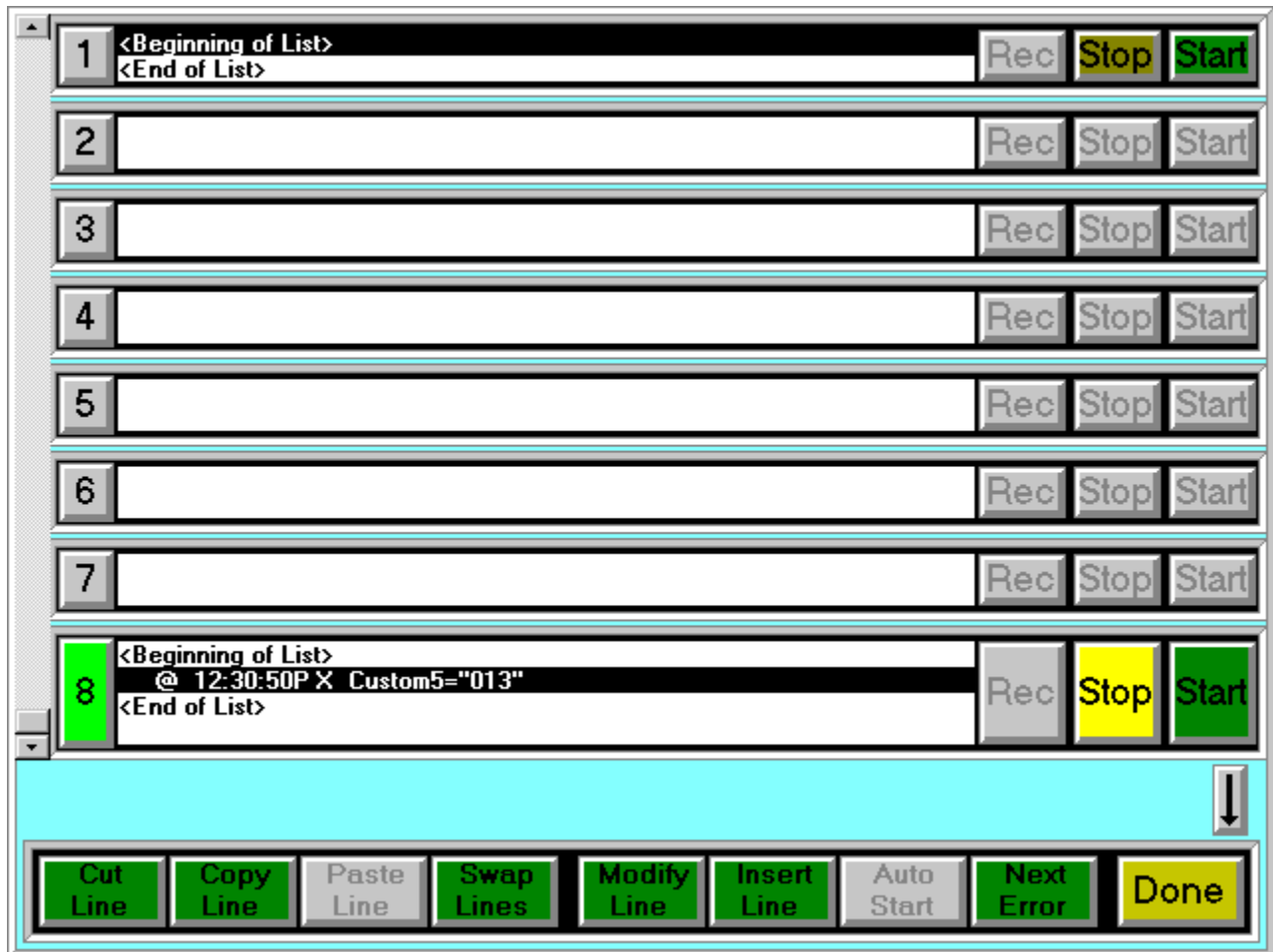
Or, "When I call for Custom5="013" use Com1 to send that information, but put a *0 at the beginning and a carriage return at the end. So my output to the switcher says *0013<CR>"

How to add a custom line to your playlist.

Continuing with the information already presented. Let's say we want to send Input 1 to Output 3 using our Custom5 command. Insert a line into your playlist that looks like the following.

The screenshot shows a dialog box titled "Insert/Modify Event". It has several input fields and a grid of buttons. The "Name/No." field contains the text "Custom5="013"". The "Time" field contains "12:30:50PM" and there is a green "Find" button to its right. The "Event ID" field is empty. Below these fields is a "Type" section with a grid of buttons: "Audio", "Stack (L)", "Chain (J)", "Comment (*)", "Label (:)", "Command (X)", and an empty button. The "Command (X)" button is highlighted in cyan. Below that is a "Start" section with another grid of buttons: "Manual", "Auto (+)", "Time (@)", and an empty button. The "Time (@)" button is highlighted in cyan. At the bottom of the dialog are two buttons: a green "OK" button and a yellow "Cancel" button.

The AVNET screen will look like this after you add the line if you didn't make any typos:



When the command is executed or "played" the switcher will take the contents of the custom line, add the static *0 and send it to the switcher, it will then send the <CR> character. Once that's done the switcher will switch.

Some troubleshooting tips:

There are some common errors too look for when using any serial remote control. Typos are the biggest cause of failure. Additionally, you should consider the items below.

Is the Windows COMM Port setup correctly?

- Is the Baud Rate (speed), bits and Flow control in Windows correct?
 - 2400 8,N,1 and None for flow control is a common configuration
 - 9600 is another common bit rate
 - Consult the switcher manual or manufacturer

Can you send the commands in Windows HyperTerminal?

- This application can be used to directly address the COMM Port
- If you can talk to it in HT you should be able to address it in the Vault

Does your IDD_Custom line address the correct COMM Port?

- Ensure that your INI reflects the COMM Port you are connected to
- Generally, this will be COM1 or COM2

Are you using the cable provided by the manufacturer and does its length exceed 25ft?

- Cables provided with serial devices may be NUL Modem rather than straight through. If you need to make a custom cable refer to the manufacturer's instructions or call their customer service division.
- It is not recommended you exceed 25ft or use extenders
- Do not run cables in parallel with power lines or fluorescent lights this can cause interference with will corrupt your data.

Some commands sent by AudioVAULT must be sent as hexadecimal code

- Some characters are reserved for system usage by AV and may not transmit correctly.
 - The Carriage Return command is one the correct syntax is backslash-X-Zero-D (\x0D) followed by a semi-colon (;)
 - The Comma is another. The correct syntax is backslash-X-2-C (\x2C) followed by a semi-colon (;)
 - Note the parentheses are used for ease of reading and not to be used in the INI file. Also the \x is a preface or "preparatory" command that lets AudioVAULT know the next two characters are hexadecimal and not to process them internally – just send them.
 - Hexadecimal conversion tables are available on the Internet from several sources and are handy to have when doing serial remote control programming.

For additional information on this topic, please contact Broadcast Electronics Digital Customer Service at 217.224.4700. You can also email specific questions to service@bdcast.com.