Beginner’s Guide to the Environment

by Len Sasso

for Version 3.0

— e —

October 1997

Logic AUDIO

emagic

Soft- und Hardware GmbH

Chapter 1

Overview of the Environment

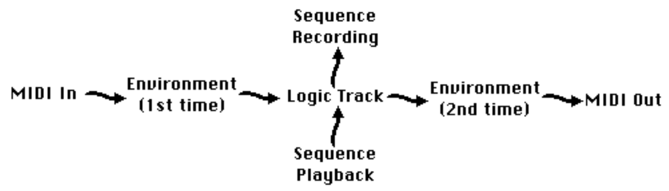


Fig. 1: MIDI signal path through the environment

First and foremost, the environment is Logic's contact with the outside world of MIDI controllers and sound devices. As *figure 1* shows, all MIDI data passes through the environment once when being recorded by Logic and again before being played pack.

Think of the environment as a virtual rack containing a patch bay and various MIDI processing modules. In environmentese, the modules are called objects and the patch cables are, conveniently, called cables. There is a little bit of normaling built in, but mostly you need to do the cabling yourself. At the cost of a little setup time you gain a system that is totally customized to your working … what? (You guessed it – "environment".)

In Logic, you come in contact with the environment in two places: 1) the Arrange window's Instrument menu which becomes visible when you click and hold on any item on the track list and 2) the Environment window. Every item (almost) on the Instrument menu is an object in the environment. There are two exceptions, named Folder and No Output. Any object in the environment can be placed on the Instrument menu, but for clarity of operation, there are usually many environment objects which do not appear there. I.e. the Instrument menu contains some but not all environment objects (and little else).

Why do environment objects appear on the Instrument menu? The MIDI data that makes up a Logic song is contained in sequences. The sequences are placed on (i.e. played by) tracks. Since the environment is Logic's conduit to the outside world, tracks need to be linked to environment objects. Prior to version 2.5.3, tracks were in fact, named after their environment object. Now tracks can have individual names but the essential link to an environment object remains.

There are many kinds of environment objects but the kind that is most often linked to a track is called an Instrument. This is why the menu that opens up when you click on a track-name is called the Instrument menu. The important thing to remember is that each track is linked to some environment object, but not necessarily an Instrument-type object. When we talk about the Instrument menu we're really talking about a menu of environment objects only some of which are Instrument-type objects.

Although every track is linked to an environment object, there is a difference between a track and its environment object. A track is a place holder in the Arrange window for containers of MIDI data (i.e. sequences and folders). A track's environment object is the destination to which this MIDI data is sent. Many tracks can send to the same environment object and as we shall see later, these tracks have a lot in common. What they do not have in common is the MIDI data they hold.

There are many uses for the environment beyond getting MIDI data into and out of Logic. MIDI data can be modified there – for example, notes can be transposed, duplicated, channelized, filtered, accented and even changed to other kinds of MIDI events. MIDI data can be created there – for example, virtual sliders can be moused to generate volume, pan, expression, after touch, mod wheel or any other kind of controller data in real-time. And, MIDI data can be time-processed there – for example arpeggiated and delayed. So, the environment is a virtual rack of MIDI processors, cables and patch points which can be as simple or complex as you make it. But, for this to have any value, MIDI data still has to get in and out of your computer.

Throughout this guide, I have referred to environment objects in small capitals to distinguish them from similarly named objects in the real world. For example, "Keyboard", always refers to the object in the environment which looks like a piano keyboard. On the other hand, "keyboard", may refer to your computer keyboard or your MIDI controller keyboard but never to the environment's Keyboard object.

1.1 MIDI Setup

Most likely, your computer is not designed to handle MIDI data and you will need a "MIDI interface" with jacks for the plugs on the MIDI cables and hardware for synchronizing the data to your computer's serial or parallel port format. MIDI interfaces often provide other functions as well – things like time code synchronization and multiple "virtual" ports. For our purposes, it is sufficient to think of the MIDI interface as providing one or more incoming and outgoing MIDI data spigots. Logic handles the management of data to and from these spigots through objects in the environment – cables to and from these objects control the flow of MIDI data that Logic records and plays back.

To follow the examples in this guide, you will need a MIDI controller keyboard and a MIDI sound device. Preferably but not necessarily the MIDI sound device should be multi-timbral (i.e. be able to play different sounds on different MIDI channels) and have a general-MIDI mode. Also, the MIDI keyboard should have a modulation controller, a foot switch, a pitch wheel and a provision for sending MIDI program changes.

You will need to have the MIDI-out from your MIDI keyboard connected to the MIDI-in of one of your MIDI interface's ports and to have the MIDI-out from one of your MIDI interface's ports connected to the MIDI-in of your MIDI sound device. If the MIDI keyboard and MIDI sound device are the same, you will also need to turn internal (local) control off. Set the MIDI channel of your MIDI keyboard to 1 and ensure that at least channel 1 on your MIDI sound device is enabled to receive MIDI data.

You can test your MIDI setup within Logic: Launch Logic and create a new song. Select the first track and, from the Instrument menu, select No Output. Select (create if necessary) a second track and, from the Instrument menu, select Modem Port. [PC: Interface] Open a transport window in Logic and locate the MIDI indicator display in the top right corner (see *figure 2*).

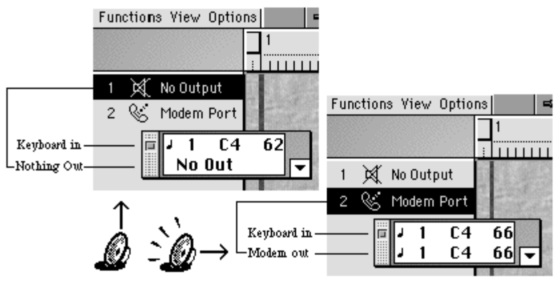


Fig. 2: Testing MIDI connections to Logic

Make sure your interface is not in "Thru" mode and select the No Output track. Now, play and hold a note on your MIDI keyboard. You should see the note displayed in the MIDI indicator but you should not hear it on your MIDI sound device. If you do hear the note, your MIDI keyboard is communicating with your MIDI sound device through some other connection (possibly internal) – search and destroy.

Next select the Modem Port [PC: Interface] track and play a note on your MIDI keyboard. This time you should hear the note on your MIDI sound device. If you do not hear the note, check the channel of your MIDI keyboard; ensure that this channel is enabled on your MIDI sound device and re-check all the MIDI connections.

1.2 In the Beginning There Were "No Output" and "Folder"

Let's start by seeing what you absolutely must have in the environment. Open a new Logic song and use **File > Save As** to save it in a convenient location with some provocative new name like "Eg". (This protects your "Autoload" song from the havoc we're about to wreak.)

Set up screen set #1 so that it contains just an Arrange window and an Environment window. Depending on your screen's size, these can be fully visible or partially overlapping. The point is to be able to easily switch back and forth between them.

Top the Arrange window and delete all but one track. (Create a track if none exists.) Click and hold on this track's name to see the Instrument menu. You'll see a bunch of junk there – just remember that it's fairly long and that the top two items are named No Output and Folder.

Now top the Environment window. On the left side of the window you'll see the parameter box (*figure 3*). (If you don't see this, select **Parameters** from the Environment window's **View** menu and it will open up.) In the **Parameters** section find and open the **Layers** menu and select the layer named **All Objects**. This shows a text-style list of everything in the environment.

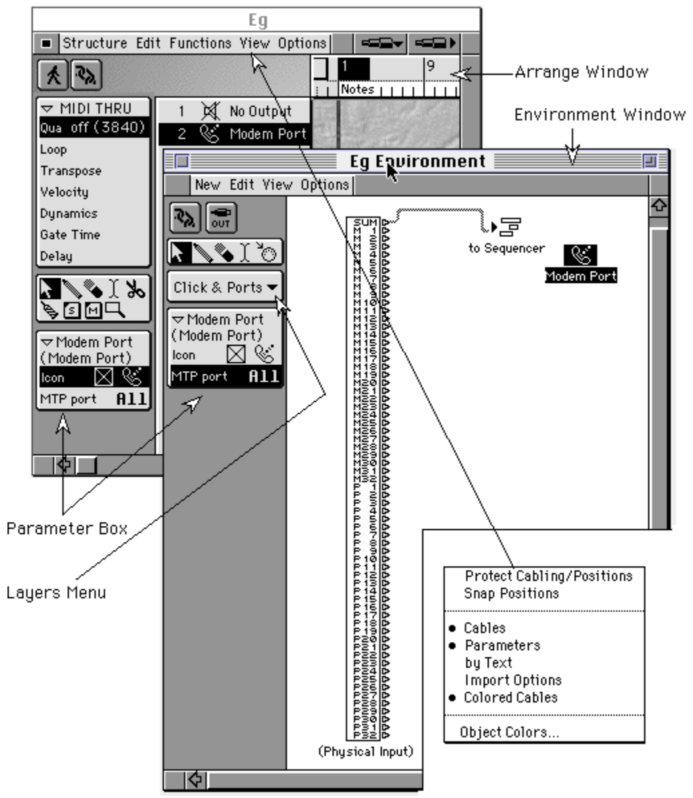


Fig. 3: Environment & Arrange windows w/ parameter box & View menu

Use Logic's **Edit** menu or a key-command (*Command-A* on the Mac) to select all environment objects. Every object on the All Objects layer should now be selected. Use Logic's **Edit** menu or the delete key to delete all these objects.

Congratulations, you have just trashed your environment. No real damage has been done though, because:

ℹ Each logic song has it's own environment.

To avoid having to recreate the environment each time you start a new song, it is customary to create an Autoload song as a starting point for new songs. Another approach is to load a song which already has an environment suitable for the new song and to delete all other song data. A third possibility is to import a suitable environment from any other Logic song. (Importing environments is discussed in detail at the end of this guide.) The last two approaches involve more steps when you start a new song but make it easier to maintain a constantly updated environment.

Go back to the Arrange window and open the Instrument menu. Notice that there are now only two items – No Output and Folder.

1.3 It Doesn't Take Much More

We are now going to create the minimum environment for running Logic. Use the environment's Layers menu to select the layer called Clicks & Ports. This layer is currently empty but we are about to add some objects to it.

Objects are created in Logic's environment by selecting them from the Environment window's New menu. From this menu select Physical Input. A tall object with lots of cable outlets should appear in the Environment window. Next select Modem Port" [PC: Interface] from this menu. A smaller object with an icon resembling a telephone handset should appear. [PC: ?] (You can check the Arrange window to see that these objects also show up on the Instrument menu.)

[PC: ?On the PC, the Physical Input object does not have multiple outputs. It has the appearance of a small MIDI plug and all data comes out of the same outlet. This outlet serves the same purpose as the SUM outlet on the Mac's Physical Input object and should be used accordingly. In version 3.0, the Physical Input will have the same appearance in both the Mac and PC.]

[PC: The environment's **New** menu on the PC has one output object named **MIDI Out Port**. This object looks like a MIDI connector (the same as the icon used for the MIDI tool or the Multi-Instrument). When created, it has the default name Interface. In its parameter box, you can select which physical port on your MIDI interface it addresses.]

With the mouse, draw a cable from Physical Input's top output, labeled Sum, to the Modem Port object. The Clicks & Ports layer of your environment should now look like *figure 4*. Play your MIDI controller keyboard and you should hear your MIDI sound device. This replicates the thru function of your MIDI interface.

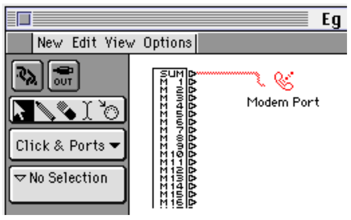


Fig. 4: Physical input and modem port objects

Top the Arrange window and select the first (and only) track. As this track's instrument, select the Modem Port [PC: Interface] from the Instrument menu. Now with the pencil tool, create a new sequence on this track. Open the sequence in a Score or Matrix editor and again with a pencil tool, put notes in the sequence. Ensure these notes are on a channel your MIDI sound device is responding to (for example, the channel your MIDI controller keyboard is using) – use the Event editor to edit the channel of the notes in the sequence if necessary. Move Logic's SPL (Song Position Locator) to the beginning of the sequence and start Logic playing. You should hear the sequence – Logic can now function as a primitive MIDI-playback device.

Delete the sequence you just created. Return the SPL to the beginning of the song and start Logic recording. Play your MIDI keyboard. You will hear what you play but, frustratingly, you will see that it's not being recorded. Back to the environment.

Top the Environment window and select **Sequencer Input** from the **New** menu. This will create a new object in the environment named to Sequencer. Click the mouse on the SUM output of Physical Input and notice its cable disconnect from the Modem Port object. While holding the mouse, draw this cable to the new to Sequencer object. Your environment should now look like *figure 3*.

Return to the Arrange window and try recording again. This time you not only hear the notes but they get recorded in a new sequence. Logic now functions as a primitive MIDI-recorder. This is the minimum environment you must have to make Logic work. It's kind of like putting a lawnmower engine in your Ferrari, but it will work.

Physical Input? To Sequencer? Huh?!?

Logic could have been created with just one kind of input. It would be a single object functioning like the two objects, Physical Input and to Sequencer, with the cable between them – everything coming into your

MIDI interface would be sent directly into Logic. This object would probably be called "Physical Input to Sequencer" and because Physical Input has no inputs and to Sequencer has no outputs, this object would have no inputs and no outputs. Having no inputs or outputs, we could eliminate it entirely. But, this would rob the environment of a lot of flexibility.

Physical Input is actually an output – it's a spigot from your MIDI interface into Logic. To Sequencer is like a hose from the environment to Logic's Arrange window. Connecting Physical Input to to Sequencer as we have done, plugs the MIDI spigot directly into the Arrange window. This is the most common configuration and is suitable for many applications. But suppose you wanted to change the MIDI data in some way before it got to the Arrange window. Suppose for example, that you didn't care for the velocity curve (feel) of your MIDI keyboard or that you wanted to transpose it and it had no controls to do so or that you wanted to split the keyboard. All of these things and much more can be done to the incoming MIDI data before it ever gets to the Arrange window simply by inserting environment objects between Physical Input and to Sequencer. That's why they're separate.

The to Sequencer has only one parameter – a checkbox named Channelize (*figure 5*). When turned on (checked), incoming MIDI data has its channel changed to the channel of the instrument for the currently selected track. If the track's instrument does not have a channel parameter then no channelizing will occur.

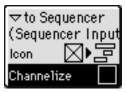


Fig. 5: The channelize parameter of the "to Sequencer" object

Let's look again at what happens to MIDI data entering Logic when the environment is configured as in *figure 3*. The MIDI data flows out of Physical Input which is like a spigot from your MIDI interface. From there it flows directly into the Arrange window through the cable connecting Physical Input to to Sequencer. What happens next depends in the Arrange window. Anything coming into to Sequencer goes to the currently selected track. The track sends it to the environment object which is selected from the Instrument menu with the mouse by clicking on the track name. (Exceptions are when No Output or Folder have been selected from the Instrument menu.) So, with Physical Input cabled directly to to Sequencer, the only path into the environment is through the tracks in the Arrange window.

Selecting a track is like connecting a temporary cable from to sequencer to the track's environment object. De-selecting the track breaks the cable.

When turned on (checked), the Channelize parameter for the to Sequence object causes incoming MIDI data to have its channel changed to the channel of the instrument for the currently selected track. If the track's instrument does not have a channel parameter then no channelizing will occur.

Let's look at our current Instrument menu choices? First, there are No Output and Folder. These choices send the data nowhere. Next there's the Modem Port [PC: Interface] object. This sends the data directly to your MIDI interface which, hopefully, sends it on to your MIDI sound device. Selecting this from the Instrument menu allows you to play your sound device from your MIDI keyboard. Then there's to Sequencer. This sends the data back into Logic – not an intelligent choice at the moment, but we will see later that it has its uses. Finally, there's the Physical Input object. This is like sending the data back out the entrance – it has no uses and this object could well be removed from the Instrument menu. So let's do it:

In the Arrange window select Physical Input from the Instrument menu. Now in the parameter box at the left side of the Arrange window you'll see a check box next to the icon (*figure 6*). Click this box so that the "X" disappears. Physical Input no longer appears on the track Instrument menu. Check it and see.

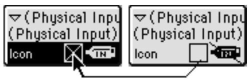


Fig. 6: Taking the Physical Input object off of the Instrument menu

1.4 Monitoring Your Progress

Go to the Environment window and select **Monitor** from the **New** menu. The Monitor is one of the simplest but most useful environment objects. It allows you to see exactly what MIDI data is coming out of any environment object that is cabled into it and for convenience, it will pass this MIDI data on (unaltered) to any object to which it is cabled.

If you start and end each environment patch with a Monitor, you can see what's happening as you are building the patch.

Click on the SUM output of Physical Input. When its cable disconnects from to Sequencer, draw the cable to the Monitor object you just created. Now cable the Monitor's output to to Sequencer. Play your MIDI keyboard and watch the note-events appear in the Monitor window (*figure 7*). Notice that each line in the Monitor window begins with an icon (note or note with slash) followed by a number (channel), a note name (e.g. C4) and another number (velocity). Move the modulation controller of your MIDI keyboard. Now the Monitor shows a small fader icon followed by three numbers (channel, controller number, controller amount). Move the pitchbend controller; step on a foot switch; make a program change. Each of these actions shows up in the Monitor.

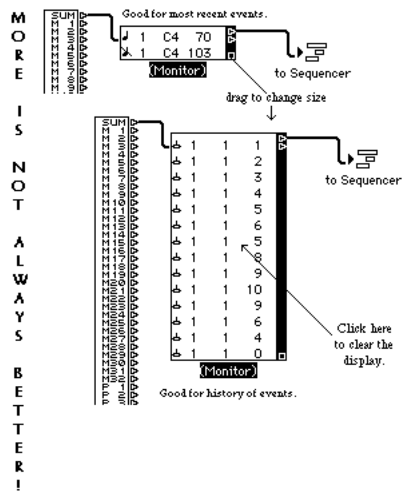

MORE IS NOT ALWAYS BETTER!
Good for most recent events.
(Monitor)
to Sequencer
drag to change size
to Sequencer
Click here to clear the display.
(Monitor)
Good for history of events.

Fig. 7: Large and small monitors between PI and "to Sequencer"

Although this connection shows the Monitor in action, it is a bit redundant – it shows the same information contained in the Transport's MIDI-in/MIDI-out window. In other locations, it becomes an invaluable debugging tool and remember, you can have more than one Monitor at a time.

1.5 A Different View

Logic's Keyboard object is another way to Monitor what's going on in the environment. Unlike the Monitor, the Keyboard only shows MIDI note events and has no history. If you use it as a track's instrument, it will show the currently held notes during sequence playback. If you select the track, it will also show currently held, incoming MIDI notes.

The Keyboard object creates notes when you click on it with the mouse. This is our first example of a way to generate MIDI data directly in the environment and can be a useful testing and debugging tool.

MIDI provides for 128 notes using note numbers 0 thru 127. This is 10 octaves plus 8 notes. Ignoring the extra 8 notes, the middle of the range is note number 60 and this is usually called "middle C". On the Keyboard object, the key marked C3 corresponds to MIDI note number 60. In the Piano score style, the ledger line between the two staves corresponds to MIDI note number 60. As *figure 8* shows, the **Display Middle C as C3** checkbox in the display preferences determines whether C3 or C4 corresponds to MIDI note number 60 in the Monitor, Event editor, Matrix editor and Keyboard parameter box. (This also applies to the Mapped Instrument which we'll discuss later.)

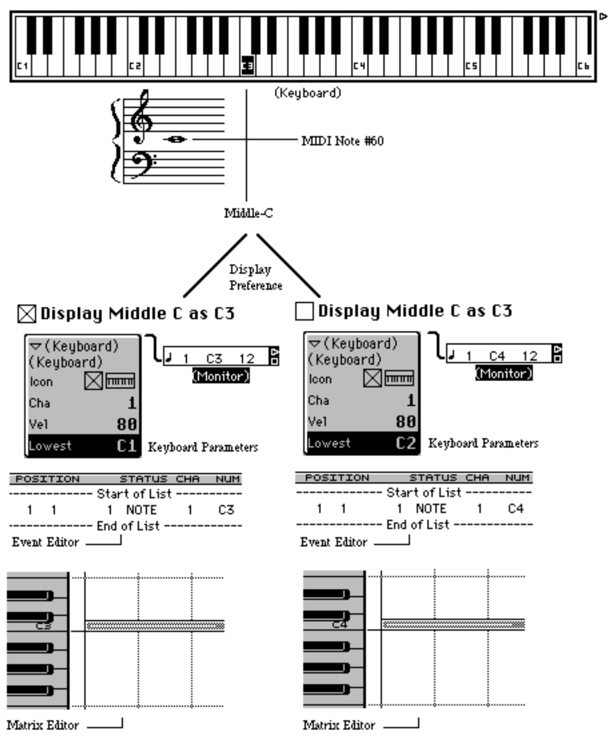


Fig. 8: Middle C display options & related note numbers

The Keyboard object's parameter box allows you to assign an icon, check whether the object appears on the Instrument menu, designate an output channel and velocity when the Keyboard is clicked (incoming data is not affected by these settings) and set the lowest note of the Keyboard display. Notice that the lowest note setting in the parameter box follows the display choice but that the Keyboard object itself always displays middle C as C3.

Middle C also has a standard musical meaning: the ledger line between treble and bass staves – the middle key on the 88-note piano keyboard – the pitch of the C below A440 (roughly 262Hz). But on the shifting sands of interconnected MIDI devices, keyboard position and pitch are the domain of the MIDI keyboard controller and MIDI sound device respectively and only the MIDI note number is available as a common reference point. You can octave-shift and transpose your keyboard so that any key plays MIDI note number 60 and you can edit your MIDI sound device presets so that MIDI note number 60 produces various pitched or non-pitched sounds but when you create or edit MIDI data in Logic, you're really working with MIDI note numbers.

Piano	Logic	Octave number	C	C#	D	Eb	E	F	F#	G	Ab	A	Bb	B
C-1	C-2	0	0	1	2	3	4	5	6	7	8	9	10	11
C0	C-1	1	12	13	14	15	16	17	18	19	20	21	22	23
C1	C0	2	24	25	26	27	28	29	30	31	32	33	34	35
C2	C1	3	36	37	38	39	40	41	42	43	44	45	46	47
C3	C2	4	48	49	50	51	52	53	54	55	56	57	58	59
C4	C3	5	**60**	61	62	63	64	65	66	67	68	69	70	71
C5	C4	6	72	73	74	75	76	77	78	79	80	81	82	83
C6	C5	7	84	85	86	87	88	89	90	91	92	93	94	95
C7	C6	8	96	97	98	99	100	101	102	103	104	105	106	107
C8	C7	9	108	109	110	111	112	113	114	115	116	117	118	119
C9	C8	10	120	121	122	123	124	125	126	127				

Table 1: MIDI Note Number s (Middle C = 60)

Chapter 2

2.1 An Instrumental Improvement

With only one output object, the Modem Port [PC: Interface], in the environment Logic's output configuration is very limited. All MIDI data whether incoming or contained in sequences is dumped, unedited, directly to your MIDI interface. Since one of Logic's primary sequencing functions is to control various MIDI sound devices, there are three environment objects designed specifically for this task and not surprisingly, they're called "instruments". The three kinds are Instrument, Multi-Instrument and Mapped Instrument.

The Instrument is intended to control a single channel of a single MIDI sound device. Select the first environment layer named Instruments – there are two such layers in the Logic default song. From the environment's **New** menu, select **Instrument** to create a new Instrument object (*figure 9*). Select this object and look at its parameters in the parameter box on the left side of the Environment window. The parameter box always shows the parameters for the currently selected object. If more than one object is selected, only the common parameters (if any) are shown.

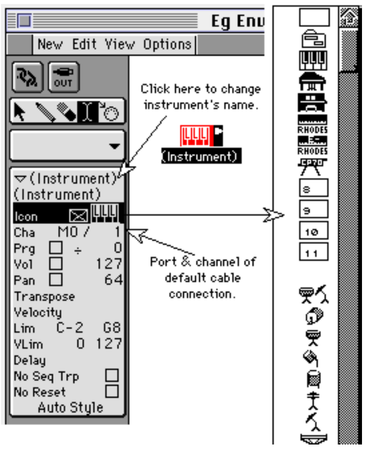


Fig. 9: Instrument object, parameter box & icon menu

At the top of the parameter box is the Instrument name, "(Instrument)" by default. You can double-click on this and name the Instrument anything you like. (When we move to the Arrange window and select this object from the Instrument menu, the Instrument name will become the default track name.) Below the name are the icon and menu checkboxes. Click on the icon to open the menu of icons that can be used for the Instrument. The choice of icons is important because when you link this Instrument object to a track in the Arrange window and give the track its own name, the icon may be the only quick way of identifying what environment object is linked to the track.

As we saw with Physical Input, the check-box next to the icon determines whether the Instrument object appears on the Instrument menu. In this case, leave it checked since the whole point of the Instrument object is for track output. We can drag & drop or use the environment's MIDI tool (MIDI icon in the tool box) to select any object as a track object even if it's not on the Instrument menu, but it is often more convenient to select it from the menu.

Now, move to the Arrange window, select a track and use the Instrument menu to select the Instrument you just created. Notice that the parameter box which appears in the Arrange window is identical to the one which appears when this object is selected in the Environment window. Settings made in either window show up in the other:

ℹ An object has only one set of parameters!

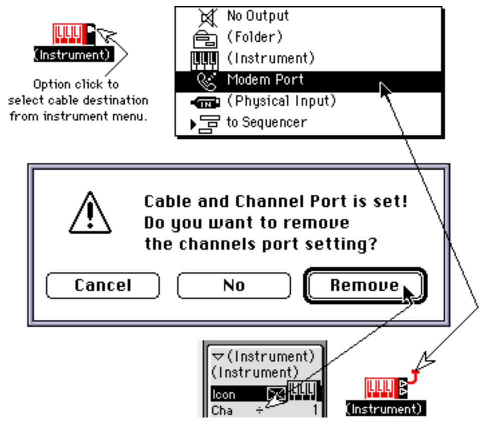
Play your MIDI keyboard. You should hear your MIDI sound device. If you don't, look just below the icon in the parameter box where you will see a port designation like M0 or P5 followed by a channel number. Ensure that the port and channel match your current setup and you will hear something.

But wait! Where are the cables?

Recall that when we created this Instrument in the Environment window, we didn't cable it to anything – it's just sitting there. As we said earlier, there is a small amount of normaling in the environment. This is it: Instruments can be cabled manually but they don't need to be – there is an automatic cabling for each Instrument which is indicated in the Instrument's parameter box. Manual cabling can either override this or be in addition to it. When an Instrument is automatically cabled, its cable outlets will be filled in, even if there is also manual cabling. If the outlets are hollow the default, automatic cabling has been removed.

Let's manually connect the Instrument to the Modem Port [PC: Interface].

Since the Modem Port is on a different layer, we'll use a short cut. Hold down the ⌥ key [PC: *Control* key] and click the Instrument's cable outlet. You will see the Instrument menu just as it appears in the Arrange window. Scroll to select Modem Port [PC: Interface]. Logic will pop up a dialog box asking if you want to remove the automatic port setting (*figure 10*). Click **Remove** and two things will happen: The port setting in the parameter box will change to "÷" indicating no automatic connection and a cable will appear leaving the Instrument. Since the cable is to an object on a different layer, only the cable beginning is shown. Want to see where it goes? Select the Instrument and from the Environment window's **Edit** menu, choose **Select Cable Destination**.



Cable and Channel Port is set! Do you want to remove the channels port setting?

Fig. 10: Recabling the instrument & "Remove" dialog box

What You See Is Not Necessarily What You Get

In addition to routing MIDI data, Instruments can also filter, modify and even create MIDI data using the Instrument's parameter box. Let's look at the available parameters (*figure 11*).

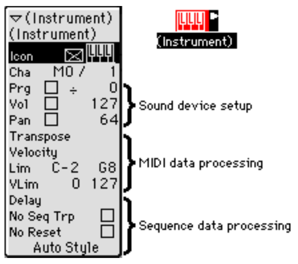


Fig. 11: Instrument parameter grouping by function

The Instrument's parameters fall into three groups. The first group is used to initialize your MIDI device's program, volume and pan settings. These values will be sent only if the corresponding box is checked – this allows you to initialize a device's program without changing its volume, for example. The values are sent whenever a track that uses this device is selected. They may also be sent whenever Logic resets depending on the **No Reset** checkbox and Logic's reset preferences settings (see below).

The second group is used to modify and/or block MIDI data. Transpose & Velocity settings are added to both MIDI input and sequence data. Lim & VLim settings block data outside the specified ranges.

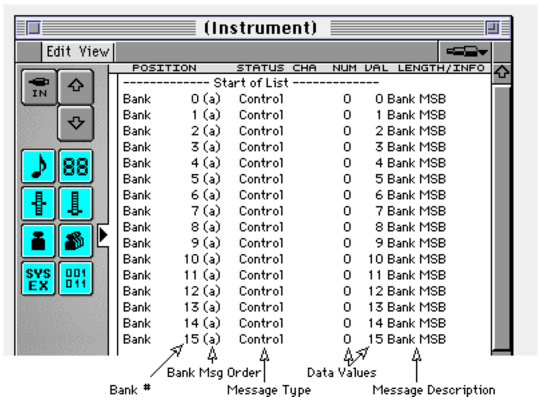
The third group affects sequences only – these parameters do not apply to MIDI input. Delay causes sequence events to be sent early or late by a set number of ticks. This is especially useful in conjunction with MIDI devices that are slow to respond or with sounds with longer attacks. No Seq Trp prevents sequence data from being transposed in spite of the sequence-parameter settings. This is intended to keep multi-sampled MIDI instruments like drums from being transposed. No Reset blocks Smart Reset messages (described below). Finally, the bottom value – Auto in the illustration – selects the initial score style when a sequence is opened in the Score editor – picking the most likely style based on note range. The score style setting can be overridden by the individual sequence parameters.

It's in the Bank

Because of the limited range of MIDI program change values (0-127), most manufacturers organize the presets of their MIDI sound devices into banks of 128 presets each. To select a preset it is then necessary to tell the MIDI sound device both which bank and which preset number. The most common way of doing this is to use MIDI controller #0 to select the bank number followed immediately by the desired program number within the bank.

Logic provides for the standard method as well as allowing you to create custom bank select messages. You specify the bank number in the Instrument's parameter window by setting the value between the Prg checkbox and the program number to the desired bank number. By default this setting is "÷" which tells Logic to not send any bank select message. Setting this to any number will cause Logic to send the standard bank select message for that bank number – controller #0 with value equal to the bank number.

To define custom bank select messages, select the Instrument in the environment, then choose **Define Custom Bank Messages...** from the **Options** menu. This will open the window shown in *figure 12* where you can define a custom message for each bank.

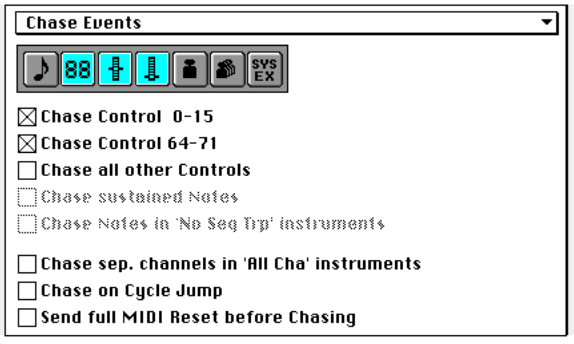


```
(Instrument)
Edit View
POSITION        STATUS CHA   NUM VAL  LENGTH/INFO
------------- Start of List -------------
Bank     0 (a)  Control      0    0 Bank MSB
Bank     1 (a)  Control      0    1 Bank MSB
Bank     2 (a)  Control      0    2 Bank MSB
Bank     3 (a)  Control      0    3 Bank MSB
Bank     4 (a)  Control      0    4 Bank MSB
Bank     5 (a)  Control      0    5 Bank MSB
Bank     6 (a)  Control      0    6 Bank MSB
Bank     7 (a)  Control      0    7 Bank MSB
Bank     8 (a)  Control      0    8 Bank MSB
Bank     9 (a)  Control      0    9 Bank MSB
Bank    10 (a)  Control      0   10 Bank MSB
Bank    11 (a)  Control      0   11 Bank MSB
Bank    12 (a)  Control      0   12 Bank MSB
Bank    13 (a)  Control      0   13 Bank MSB
Bank    14 (a)  Control      0   14 Bank MSB
Bank    15 (a)  Control      0   15 Bank MSB
```

Fig. 12: Custom bank message window

Chase, Reset or Forget?

When you start playback from the middle of a song, you may want your MIDI devices updated with the controller, pitch bend and program values that would have been most recently sent, had you played from the beginning of the song. This function is called "Chase" and the **Chase Events** section of the **Song Settings** (*figure 13*) allows you to specify what MIDI data to chase. Chasing events can cause a lot of MIDI data to be sent thus causing a hiccup in timing. This is especially annoying on cycle jumps. As a general rule, chase as little as possible.



Chase Events

- [x] Chase Control 0-15
- [x] Chase Control 64-71
- [ ] Chase all other Controls
- [ ] Chase sustained Notes
- [ ] Chase Notes in 'No Seq Trp' instruments
- [ ] Chase sep. channels in 'All Cha' instruments
- [ ] Chase on Cycle Jump
- [ ] Send full MIDI Reset before Chasing

Fig. 13: Song settings window - chase events section

Reset messages are a special kind of chase-event – rather than sending the most recent value, they send a standard or neutral value. The events to which reset applies are:

Smart	MIDI Event	Reset Value
	All Notes Off (Controller 123)	0
	Reset Controls (Controller 121)	0
√	Sustain Pedal (Controller 64)	0
	Foot Control (Controller 4)	0
	Breath Control (Controller 2)	0
√	Modulation (Controller 1)	0
√	Channel Pressure	0
√	Pitch Bend	Centered

There are four conditions that will cause Logic to send reset messages:

- Pressing the "Stop" key twice.
- Clicking the MIDI out display on the Transport.
- Pressing the full panic key-command.
- Opening or activating a new song.

Logic can handle resets in one of two ways: "smart" and "not?so?smart". When you turn a reset message on by X-ing its checkbox in the **Reset Messages** section of the **Preferences** window, Logic resets that controller for every channel on the X'd port. This is the not?so?smart method – it results in a lot of unnecessary MIDI data. For pitch bend, channel pressure, modulation and the sustain pedal, Logic will determine for each track in the Arrange window, whether the last (if any) value sent was different than the reset value and only then will Logic send a reset message to the port and channel of that track. This is the smart method – only the necessary MIDI data is sent. X-ing the checkbox of one of these event types in the Reset Messages dialog, stupefies the smart reset – don't do it. X the other event types only if necessary.

This brings us to the **No Reset** Instrument-parameter. When it is X'd, Logic does not send smart resets to the port and channel of this Instrument. (Note that if there is another track's Instrument with **No Reset** not X'd and using the same port and channel, smart resets will be sent.)

Reducing Instrument Clutter

So far, we've seen how the Instrument may be used to control one channel of a MIDI sound device. You probably have several sound devices and some of them may be multi-timbral – able to play different sounds on different MIDI channels. When you create a complex song, you are going to want an Instrument for each channel of each MIDI sound device you use in the song. There is another Instrument object, the Multi-Instrument, designed to reduce the number of separate objects you need to accomplish this.

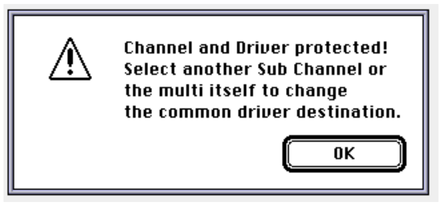
The Multi-Instrument is really 16 Instruments rolled into one. Why 16? Because a MIDI port carries information for 16 MIDI channels and most multi-timbral MIDI sound devices are connected to one port and support up to 16 separate sounds. Logic's Multi-Instrument is intended for controlling multi-timbral MIDI sound devices but it is equally well suited for controlling single channel MIDI sound devices.

One important advantage of using Multi-Instruments is that they provide for preset names. When you're working on a song and want to choose a sound for a particular part, it's obviously much easier to choose from a list of names than simply by number. Squinting to read your synth's LCD across the room while scrolling thru program numbers is decidedly unfriendly. If you use a Multi-Instrument for a single channel MIDI sound device, you can select programs by name. Multi-Instruments will even hold sets of preset names for up to 15 different banks. But, this comes at a price – Multi-Instruments take up significantly more memory than Instruments and as a result, your songs get bigger. So, when an standard Instrument will do, use it.

Logic provides a handy trick for combining Instruments and Multi-Instruments. If an Instrument is cabled to a Multi-Instrument, the Instrument's program menu will show the Multi-Instrument program names. (It even works if the cable passes through a couple of other objects – the limit seems to be 2). This allows you to have separate environment objects for individual Multi-Instrument channels without having several Multi-Instruments containing the same program names. Note that when using an Instrument cabled to a Multi-Instrument, the program names do not appear in the Arrange window's track list – you must use a sub-instrument of a Multi-Instrument for this.

Go to the Environment window and on the same layer as the Instrument you created, choose **Multi-Instrument** from the **New** menu. A Multi-Instrument object will appear with the default name "(Multi Instr.)".

The 16 numbered squares on the Multi-Instrument object correspond to the 16 individual sub-instruments contained in the Multi-Instrument. Each sub-instrument has its own set of parameters and these are exactly the same as for the Instrument discussed above. The one exception is that the port and channel parameters can not be changed – if you try you will get a warning dialog (*figure 14*). What this means is that the port is controlled by the Multi-Instrument's global settings and the channel is fixed for each sub-instrument – the channel is same as the sub-instrument number.



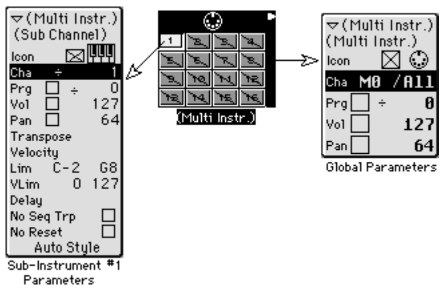
Channel and Driver protected!
Select another Sub Channel or
the multi itself to change
the common driver destination.

Fig. 14: Dialog when trying to change sub-instrument channel

You can change the channel setting from the Arrange window's parameter box – the effect here is to change the selected track's instrument to a different sub-instrument.

Select the Multi-Instrument by clicking in the top region of the object, somewhere near the MIDI connector icon. Now look at the parameter box (*figure 15*) and you will see an abbreviated set of parameters: the familiar icon and Instrument menu checkbox, the port and channel settings and the program, volume and pan checkboxes and settings. The icon and its checkbox work as usual – you can use the Multi-Instrument itself (rather than one of the sub-instruments) as a track's instrument. The port setting is global for all the sub-instruments. The channel setting is also global – setting it to a channel number rather than its default setting of All will cause all sub-instruments to send to the same channel. You probably don't want this so leave it set to All. Finally, the program, volume and pan settings, if X'd, will result in values being sent to the Multi-Instruments channel (or to channel 1 if this is set to All). You probably want to leave these unchecked also.

Leave the Multi-Instrument's global parameters at their default settings and set the sub-instrument's parameters as you would an instrument.

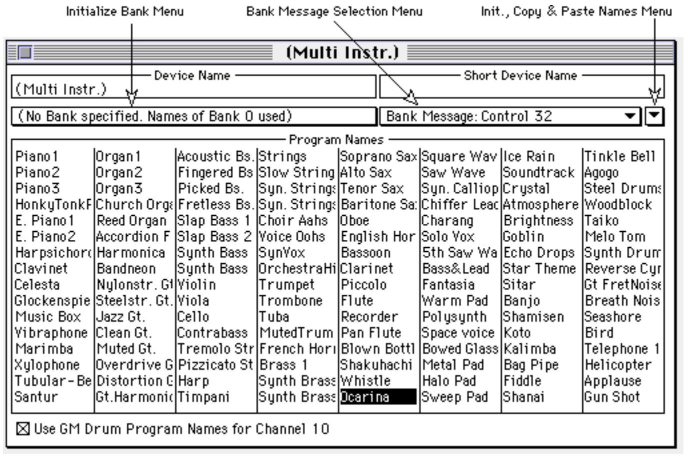


```
▽(Multi Instr.)
 (Sub Channel)
Icon
Cha      ÷       1
Prg  ☐   ÷       0
Vol  ☐         127
Pan  ☐          64
Transpose
Velocity
Lim    C-2     G8
VLim     0    127
Delay
No Seq Trp      ☐
No Reset        ☐
    Auto Style
```

```
▽(Multi Instr.)
(Multi Instr.)
Icon
Cha  M0  /All
Prg ☐  ÷    0
Vol ☐     127
Pan ☐      64
```

Fig. 15: Parameter boxes for multi-instrument and sub-instruments

Double clicking on the Multi-Instrument opens a window for entering preset names (*figure 16*). It comes filled with the General MIDI (GM) standard preset names. You can edit this list in various ways: you can initialize the names to GM standard or to program numbers; you can enter individual names by double-clicking on a name; you can cut, copy & paste to and from your favorite text editor [PC:?]; you can copy & paste from Sound Diver and under just the right circumstances, you can automatically update the names from Sound Diver using AutoLink.



(Multi Instr.)

Device Name: (Multi Instr.)

Short Device Name:

(No Bank specified. Names of Bank 0 used)

Bank Message: Control 32

Program Names

Piano1	Organ1	Acoustic Bs.	Strings	Soprano Sax	Square Wav	Ice Rain	Tinkle Bell
Piano2	Organ2	Fingered Bs	Slow String	Alto Sax	Saw Wave	Soundtrack	Agogo
Piano3	Organ3	Picked Bs.	Syn. Strings	Tenor Sax	Syn. Calliop	Crystal	Steel Drums
HonkyTonkP	Church Orga	Fretless Bs.	Syn. Strings	Baritone Sa:	Chiffer Leac	Atmosphere	Woodblock
E. Piano1	Reed Organ	Slap Bass 1	Choir Aahs	Oboe	Charang	Brightness	Taiko
E. Piano2	Accordion F	Slap Bass 2	Voice Oohs	English Hor	Solo Vox	Goblin	Melo Tom
Harpsichor	Harmonica	Synth Bass	SynVox	Bassoon	5th Saw Wa	Echo Drops	Synth Drum
Clavinet	Bandneon	Synth Bass	OrchestraHi	Clarinet	Bass&Lead	Star Theme	Reverse Cyr
Celesta	Nylonstr. Gt	Violin	Trumpet	Piccolo	Fantasia	Sitar	Gt FretNoise
Glockenspie	Steelstr. Gt.	Viola	Trombone	Flute	Warm Pad	Banjo	Breath Nois
Music Box	Jazz Gt.	Cello	Tuba	Recorder	Polysynth	Shamisen	Seashore
Vibraphone	Clean Gt.	Contrabass	MutedTrum	Pan Flute	Space voice	Koto	Bird
Marimba	Muted Gt.	Tremolo Str	French Horn	Blown Bottl	Bowed Glass	Kalimba	Telephone 1
Xylophone	Overdrive G	Pizzicato St	Brass 1	Shakuhachi	Metal Pad	Bag Pipe	Helicopter
Tubular-Be	Distortion G	Harp	Synth Brass	Whistle	Halo Pad	Fiddle	Applause
Santur	Gt.Harmonic	Timpani	Synth Brass	Ocarina	Sweep Pad	Shanai	Gun Shot

☒ Use GM Drum Program Names for Channel 10

Fig. 16: Window for entering MI preset names

Checking the checkbox at the bottom of the Multi-Instrument window will cause the channel 10 sub-instrument to behave differently than the rest. Instead of showing the program names in the Multi-Instrument window, it will show the GM-standard drumset names.

ℹ You can copy and paste names between the multi-instrument window and any text editor for easier editing. Names can also be pasted to the multi-instrument from Sound Diver.

You can use the Multi-Instrument's preset name window to select presets on your MIDI sound device. In order for this feature to be enabled, the Prg checkbox for the desired sub-instrument must be checked. Then, double-clicking a sub-instrument's number and clicking names in the preset name window will cause program change messages to be sent to the channel of that sub-instrument. (Using the global instrument for this when its channel is set to ALL will result in program changes on channel 1.) Whether a bank message is also sent depends on the global Prg bank indicator – a "÷" indicates no bank message and a number specifies which bank message. The selected track in the Arrange window has no effect on this process and neither does enabling or selecting sub-instruments.

ℹ There are two convenient tricks for opening and using the preset name window from the Logic's Arrange window. If you click on the Multi-instrument's name in the sub-instrument's parameter box, the window will open for selecting presets on the sub-instrument's channel. Alternatively, holding the Option or Control keys and double click the track name will also open the preset name window.

2.2 Getting a Multi-Instrument to Speak

What follows assumes you have a multi-timbral MIDI sound device with a GM mode and that the Multi-Instrument's global port setting matches the port to which this device is connected. If you do not have such a MIDI sound device, you will need to adapt the information to your setup.

When you first create a Multi-Instrument, each of the 16 squares corresponding to the 16 sub-instruments has a slash through it. This indicates that the sub-instrument is not activated – i.e. its icon checkbox is not X'd and it does not show up on the track list's Instrument menu. When you click on one of these squares on the Multi-Instrument object, the slash disappears and the sub-instrument's parameter box opens with the icon checkbox X'd. This is how you activate a sub-instrument. You can deactivate it by clicking the icon checkbox thus un-X'ing it. Note that this only happens the first time you click on a sub-instrument number. Once you've deactivated a sub-instrument, you must use its icon checkbox to reactivate it. There is a good reason for this:

The purpose of activating and deactivating sub-instruments is as with all other environment objects, to control Instrument menu clutter. It has no other effect on the sub-instrument and indeed, you can drag a sub-instrument's numbered square to the track list and then use it as a track's instrument without ever activating it. This is the reason that activation is not automatic after the first time – you may want to use sub-instruments without putting them on the Instrument menu. (Another method of getting a sub-instrument on the track list is to first select the desired track then click the sub-instrument with the MIDI tool in the Environment window.)

Putting a sub-instrument on the track list does not enable the MIDI sound device. You must also put your Midi sound device in multi-timbral mode and enable its various channels.

In general when using a Multi-Instrument, you will want to deactivate the Multi-Instrument display and you will only want to activate those sub-instruments for enabled channels on the MIDI sound device. You may want to use a Multi-Instrument for single channel MIDI sound devices in order to have preset names rather than numbers. In this case, you still want to deactivate the Multi-Instrument and activate the sub-instrument for the desired channel. This way you will have a full set of Instrument parameters rather than the abbreviated set associated with the Multi-Instrument. As mentioned above, you could also use an Instrument cabled to the Multi-Instrument.

Let's first use the Multi-Instrument itself (not one of its sub-instruments) as a track instrument either by selecting it from the Instrument menu or dragging it to the track list. Your Arrange window should look something like *figure 17*. One quick way to check who's listening on this port is to scroll the Multi-Instrument's channel (shown as All here) through the 16 channels while playing your MIDI keyboard. (You could also do this with an Instrument.) If you don't hear anything on a particular channel, this could be for several reasons:

- The channel may not be enabled on the receiving MIDI device.
- The sound may be muted at your mixer.
- The channel preset's volume may be too low.

- The channel preset may be silent (e.g. a default “blank” preset).

The last two of these can be checked by temporarily enabling Prg and Vol on the Multi-instrument and adjusting their values.

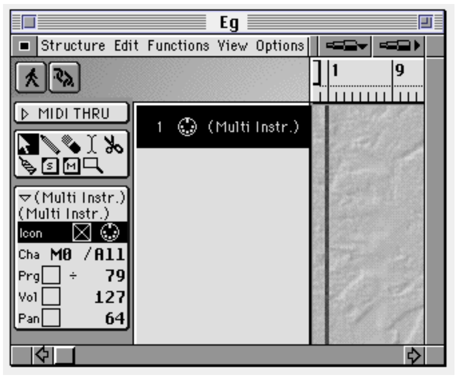


Fig. 17: Multi-instrument selected for Arrange window track

A Multi-Instrument and its 16 sub-instruments can each have their own icon. In addition, if the Prg checkbox is X’d, the track list will show the short name of the Multi-Instrument followed by the preset name. (The short name is blank by default but can be changed in the upper-right region of the Multi-Instrument window.)

The Multi-Instrument comes equipped with a menu of bank-select options. When the Multi-Instrument is created this is set to us MIDI controller #32 for bank select – use the menu to select any of the other options. One of these options is *Custom Bank Messages* and this is exactly the same as for Instruments (described above).

Multi-Instruments provide for 15 separate banks of preset names. These are intended to match the banks of the associated MIDI sound device. Banks other than the first bank need to be initialized. You initialize a bank simply by selecting it from the bank menu in the Multi-Instrument window. Things to remember:

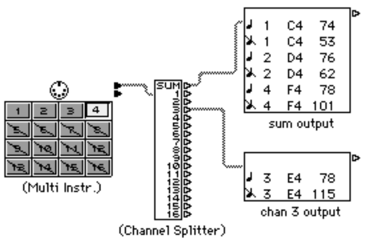
- If you initialize a new bank, all lower numbered banks are automatically initialized.
- All un-initialized banks use the preset names of the previous initialized banks.
- A bank’s preset names are shown when that bank number is selected in the parameter box.

Each initialized Multi-Instrument name-bank takes up additional song memory. Don't initialize banks unless you need to.

Separating Sub-Instrument Output

Notice that a Multi-Instrument has only a single cable outlet. If you connect this to something, a Monitor object for example, a new outlet will appear. If you connect this new outlet to something, a third outlet will appear. If you select a track that uses the Multi-Instrument or any of its sub-instruments and play your MIDI keyboard, all the data will come out all of the Multi-Instrument's cable ports. There is no way, using just the Multi-Instrument, to separate the outputs of the sub-instruments. But there is an environment object designed just for such tasks.

From the **New** menu in the Environment window, select **Channel Splitter**. A tall object with 17 cable outlets will appear (*figure 18*). While the object is selected, ensure that the icon checkbox in its parameter box is unchecked. (This is a good habit to get into whenever creating a new environment object that is not intended as a track instrument – it greatly reduces Instrument menu clutter and it's much easier to do it when the object is created than later. Some objects, like the Channel Splitter, have this box unchecked by default – others, like the Instruments have it checked.)



```
1   C4   74
1   C4   53
2   D4   76
2   D4   62
4   F4   78
4   F4  101
```

```
3   E4   78
3   E4  115
```

Fig. 18: Channel Splitter illustrated w/ monitors

The Channel Splitter has an outlet for each MIDI channel as well as an outlet at the top labeled SUM. Everything that does not go out one of the channel outlets will go out the SUM outlet. (E.g. if you cable the channel 3 outlet to another environment object then all data from the channel 3 sub-instrument will go out there and everything else will go out the SUM outlet.) This is very handy for applying separate environment processes to different sub-instruments.

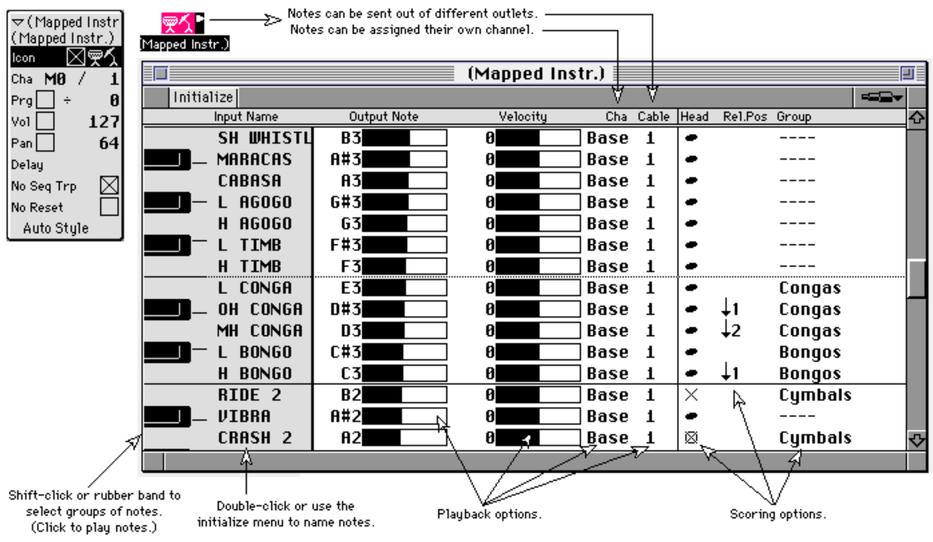
The Channel Splitter has many useful applications. Another is splitting Physical Input before it gets to to Sequencer. For example, if you have merged the data from two MIDI controller devices, say a fader-matrix and a keyboard, you might wish to send the keyboard to to Sequencer but pass the fader values directly to the output. Physical Input has separate port cables but all channel data for a single port is combined. Use the Channel Splitter.

2.3 A Special Instrument for Drums

Logic has one more kind of Instrument and it is designed especially for use with drum sequences. What makes a drum sequence different is that different notes produce different sounds rather than different pitches of the same sound. Once a MIDI note message is sent to your MIDI drum-sound device, the device decides which sound to play. It does so by using a note-map – a list of MIDI note numbers and their corresponding sounds. When creating or editing a drum sequence, it would be nice to work with drum-sound names (e.g. snare, sidestick, kick, etc.) rather than pitches. This is one of several things a Mapped Instrument does for you – it allows you to give each note a name. These names are used in the Matrix, Event and Hyper-editors to facilitate drum sequence editing.

The Hyper-editor's "Auto Define" feature allows you to easily define a drum-editing Hyper set by selecting individual drum notes in any sequence.]

The first thing that happens when you create a Mapped Instrument using the environment's New menu is that the Mapped Instrument window appears (*figure 19*). This is where you set up the individual note characteristics, but before going into that, notice that the Mapped Instrument's parameter box does not have the Transpose, Velocity, Lim or VLim parameters of the Instrument. The reason is that these parameters can be set on a note-by-note basis for Mapped Instruments. See also that the No Seq Trp checkbox is X'd by default for the Mapped Instrument. When using Mapped Instruments for percussion sequences, you generally do not want to transpose the whole sequence. For other applications, you may want to re-enable this parameter.


Notes can be sent out of different outlets.
Notes can be assigned their own channel.
Mapped Instr.)
Initialize
Shift-click or rubber band to select groups of notes. (Click to play notes.)
Double-click or use the initialize menu to name notes.
Playback options.
Scoring options.

▽(Mapped Instr
(Mapped Instr.)

Icon	☒
Cha	M0 / 1
Prg ☐	÷ 0
Vol ☐	127
Pan ☐	64
Delay	
No Seq Trp	☒
No Reset	☐
Auto Style	

(Mapped Instr.)

Input Name	Output Note	Velocity	Cha	Cable	Head	Rel.Pos	Group
SH WHISTL	B3	0	Base	1	●		----
MARACAS	A#3	0	Base	1	●		----
CABASA	A3	0	Base	1	●		----
L AGOGO	G#3	0	Base	1	●		----
H AGOGO	G3	0	Base	1	●		----
L TIMB	F#3	0	Base	1	●		----
H TIMB	F3	0	Base	1	●		----
L CONGA	E3	0	Base	1	●		Congas
OH CONGA	D#3	0	Base	1	●	↓1	Congas
MH CONGA	D3	0	Base	1	●	↓2	Congas
L BONGO	C#3	0	Base	1	●		Bongos
H BONGO	C3	0	Base	1	●	↓1	Bongos
RIDE 2	B2	0	Base	1	×		Cymbals
VIBRA	A#2	0	Base	1	●		----
CRASH 2	A2	0	Base	1	⊠		Cymbals

Fig. 19: Mapped instrument window and parameters

In addition to naming notes, you can change the output note, define a velocity offset, assign the note to a specific channel and cable and determine how it will appear in the score. Let's look at each of these characteristics.

Recall that a MIDI note message contains three pieces of information – note-number (0-127), MIDI channel (0-15) and note-velocity (0-127). Logic shows you pitch instead of note number with C-1 corresponding to note-number 0 and G9 corresponding to note-number 127. (If you select Yamaha numbering in Logic's preferences, the pitch names range from C-2 to G8.) The Mapped Instrument allows you to map any input note whether coming from your MIDI keyboard or from a Logic sequence, to any output note. In essence, the Mapped Instrument changes the note-number of notes passing through it. (As we'll see below, you can also do this using the Transformer object's map mode but using the Mapped Instrument's window is simpler and more graphic.)

When you create a Mapped Instrument, its map is set up to make no changes – the in-note number and the out-note number are the same. Why would you want to change this? Suppose for example, that you want to play a snare roll live from your MIDI keyboard. Mapping adjacent keys to the same output note makes this easy. As we go along we will see many other uses for note mapping and several other ways to achieve it?

The Mapped Instrument's velocity setting allows you to change incoming note velocities by a positive or negative amount. There is a separate velocity offset for each note. This is useful for balancing percussion accents and sound levels.

The individual channel and cable settings for each note allow you to divide the Mapped Instrument's output among several MIDI channels or among several of the Mapped-Instrument's cables for individual environment processing. You could, for example, divide a sequence among several MIDI sound devices or pass a snare sound through an Arpeggiator to get an automatic roll.

Finally, the three scoring options – Head, Rel. Pos and Group – determine how drum sequences will appear in the Score editor when a drum-type style is selected. (A drum style is one whose name is preceded by "#". For normal styles, drum sequences appear just like other sequences – the in-notes determine their staff position.) When you use a drum style, the group determines which voice the notes appear in (similar to channel for a standard style) and the Rel. Pos determines the spacing relative to the top staff line.

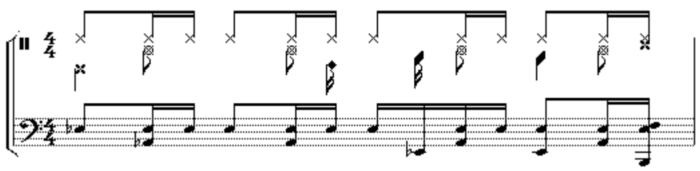


Fig. 20: Drum and normal scoring of drum tracks

2.4 Maps, Zones and Chords

Aside from playing percussion parts from piano-style keyboards, there are other reasons to redefine the relation between keyboard and note played. Some are:

- Keyboard splitting and redesigning
- Key-correcting & scale-filtering
- Automating chording & harmony
- Multi-timbral instrumentation

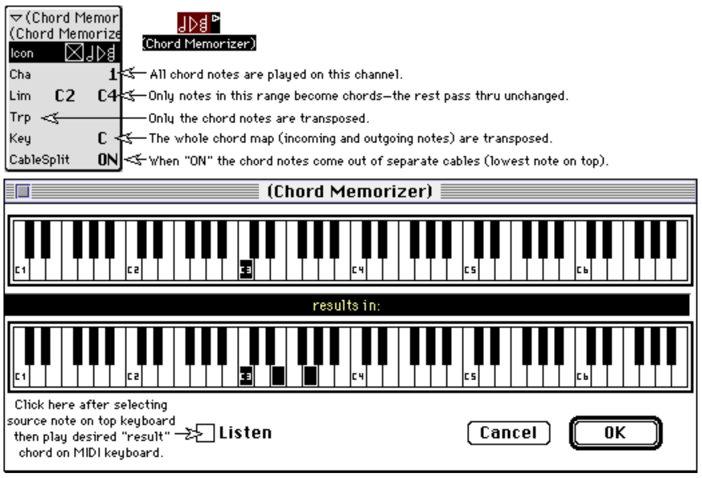
Besides the Mapped Instrument, the environment contains two other objects for accomplishing these tasks. One is the Transformer, perhaps the most versatile object in the environment. We will discuss the Transformer in detail a little later. The other object is the Chord Memorizer which, as its name implies, is intended to map single notes to chords.

ℹ Remember that "chord" can mean zero, one or several notes. This makes the chord memorizer ideal for scale-correcting and scale-filtering as well as for playing chords.

Choose **Chord Memorizer** from the environment's **New** menu then double click on the Chord Memorizer object. A double keyboard window will open (*figure 21*). This is where you map in-notes to out-notes. First select the in-note on the top keyboard then select the out-note(s) on the bottom keyboard. You can toggle out-notes on and off on the bottom keyboard as long as the in-note remains selected on the top keyboard. Notice that you can toggle all the out-notes off, mapping the in-note to nothing.

The Chord Memorizer's note map is only for one octave. You can choose an in-note in any octave and the relation between its octave and the out-note(s) octave(s) will be preserved but all other in-notes in the same pitch-class will be mapped in the same way. This allows chords to be transposed to different octaves but restricts the Chord Memorizer to 12 individual chords.

The Chord Memorizer's channel parameter sets the channel of all notes coming out of it. The Lim parameter determines which incoming notes are mapped and which are left unchanged. Notes outside of the Lim range are not mapped by the Chord Memorizer – they play through as though the Chord Memorizer were not there.



All chord notes are played on this channel.
Only notes in this range become chords—the rest pass thru unchanged.
Only the chord notes are transposed.
The whole chord map (incoming and outgoing notes) are transposed.
When "ON" the chord notes come out of separate cables (lowest note on top).
Click here after selecting source note on top keyboard then play desired "result" chord on MIDI keyboard.

Fig. 21: Chord memorizer window and parameters

The Chord Memorizer's Trp and Key parameters often cause confusion. Trp affects only the out notes while Key affects both the in and out notes. Suppose only one chord, CÆC major triad, is defined with no Trp and with the Key set to C. As *figure 22* illustrates, changing the Trp to 5 will cause C to play an F major triad while changing the Key to F will cause an F to play an F major triad. (Changing both will cause an F to play a Bb major triad.)

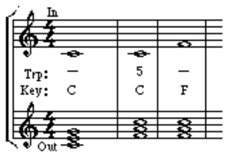


Fig. 22: Chord memorizer transpose and key parameters

The CableSplit parameter allows individual notes in chords generated by the Chord Memorizer to be split among separate cables. When this parameter is ON the lowest chord note will appear at the top outlet and the remaining notes will appear at successively lower outlets with the highest note appearing at the bottom outlet. If there are more notes than outlets, the extra notes will appear at the bottom outlet. If there are fewer notes than outlets, nothing will appear at the extra outlets.

Using the Chord Memorizer

There are several ways to connect a Chord Memorizer. The simplest is to use it as a track's instrument and cable its output directly to an output port. MIDI input or sequence playback on this track will result in MIDI output of the corresponding out notes. All output will be channelized according to the Chord Memorizer's channel parameter and incoming notes outside the Lim range will pass through unchanged (i.e. will not be mapped to out notes). Sometimes it's useful to cable the Chord Memorizer into one or more Instruments rather than directly to MIDI output. This is especially useful when the Chord Memorizer is in cable split mode. Finally, there are applications when you might want to cable an Instrument into the Chord Memorizer – for example, when you always want the output of that Instrument to follow the Chord Memorizer's map.

2.5 Running Status of the Human Kind — Part I

Before moving on to the Transformer object let's take a brief look at what we have and what we can do with it. First, we have objects for getting MIDI data in and out of Logic. These include Physical Input, to Sequencer and the port [PC: interface] objects. Physical Input is like a spigot for all the MIDI-in ports on your MIDI interface. The port [PC: interface] objects are like a hose to your computer's MIDI interface – your MIDI interface determines where the data goes after it leaves the hose. To Sequencer is a connection from the environment to Logic's Arrange window – incoming data is routed to the currently selected track's instrument.

The next objects we have are the three types of Instruments. Instruments are intended for routing MIDI data to specific MIDI sound devices and the Instrument types are tailored for the single-channel, multi-channel and mapped MIDI sound devices. Instruments can store initial program, volume and pan values. They can also change the pitch, velocity and timing of incoming notes and they can set pitch and velocity ranges outside of which notes will be blocked.

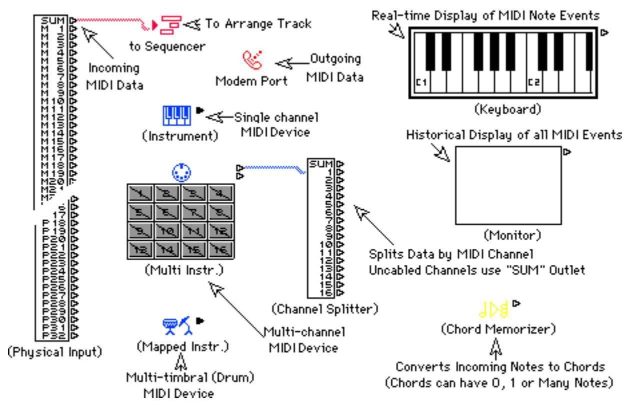

To Arrange Track
to Sequencer
Incoming MIDI Data
Outgoing MIDI Data
Modem Port
Real-time Display of MIDI Note Events
(Keyboard)
Single channel MIDI Device
(Instrument)
Historical Display of all MIDI Events
(Monitor)
Splits Data by MIDI Channel
Uncabled Channels use "SUM" Outlet
(Multi Instr.)
(Channel Splitter)
Multi-channel MIDI Device
(Chord Memorizer)
(Mapped Instr.)
(Physical Input)
Multi-timbral (Drum) MIDI Device
Converts Incoming Notes to Chords
(Chords can have 0, 1 or Many Notes)

Fig. 23: Running status I - illustration of objects so far

Then we have Channel Splitter and Chord Memorizer – both of which affect the data passing through them. The Channel Splitter routes the data according to its intended MIDI channel. Data for channels without specific cables is sent to the SUM outlet. The Chord Memorizer maps incoming notes to none or more outgoing notes. The outgoing notes are determined by the map defined in the Chord Memorizer's window.

Finally we have Keyboard and Monitor. Both are used for viewing MIDI activity anywhere in the environment. The Keyboard shows only MIDI note-events and only shows those while they are being held. The Keyboard will also generate MIDI note-events. The Monitor shows all MIDI activity and keeps an historical record until it is cleared by clicking inside its window.

What's It Good For?

These objects are enough to create a simple operating environment and even to do some interesting processing. The environment in *figure 24* addresses two output ports through two Monitor objects. The Monitors serve two purposes – they show what data is going to each port and they "isolate" the environment so that it can be easily moved to another song and connected to the output port objects in that song's environment. (Moving environments between songs is discussed at the end of this guide.)

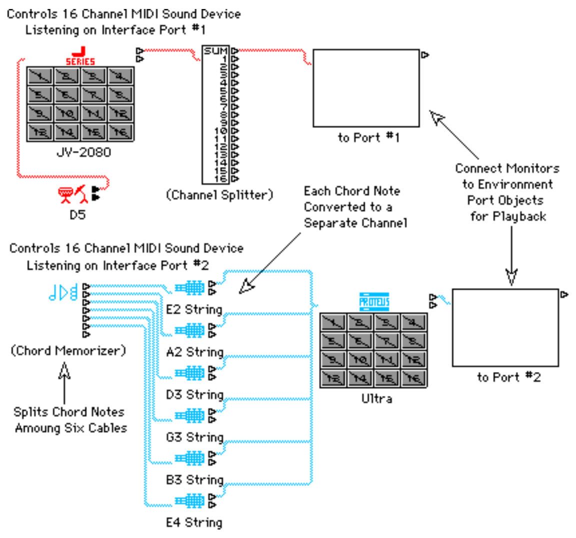

Controls 16 Channel MIDI Sound Device Listening on Interface Port #1
JV-2080
D5
SUM
(Channel Splitter)
to Port #1
Connect Monitors to Environment Port Objects for Playback
Each Chord Note Converted to a Separate Channel
Controls 16 Channel MIDI Sound Device Listening on Interface Port #2
(Chord Memorizer)
Splits Chord Notes Amoung Six Cables
E2 String
A2 String
D3 String
G3 String
B3 String
E4 String
Ultra
to Port #2

Fig. 24: Beginning Setup possible with these objects

The Multi-Instrument is intended for controlling a multi-timbral MIDI sound device. Each of its sub-instruments except #10 has been enabled (icon checked) so that it will show up on the Instrument menu. The output of the Multi-Instrument is passed through a Channel Splitter so that the individual MIDI channels could be routed and processed independently. For our purposes, the Channel Splitter's SUM output carries all channels to the Monitor which feeds the first port.

Instead of enabling the channel 10 sub-instrument, a Mapped Instrument using channel 10 has been cabled into the Multi-Instrument. The reason for this is that channel 10 is commonly reserved for drum and percussion parts and these are most easily edited and scored using a Mapped Instrument. Since the Mapped Instrument is cabled into the Multi-Instrument, the Multi-Instrument's preset names will show up in the Mapped Instrument's program pop-up menu for easy selection of the desired percussion preset.

The purpose of the patch leading to the second port is to channel the Chord Memorizer's output to six separate MIDI channels – the channels being determined by the Instruments cabled between the Chord Memorizer and the Multi-Instrument. This could be used, for example, to channel six note guitar chords to six different guitar presets – one for each string. The reason for the Multi-Instrument is to provide names to the Instruments' program pop-up menus. From the Multi-Instrument the MIDI data passes through the Monitor to the desired MIDI port.

The six cables from the Chord Memorizer carry the six chord notes in bottom-to-top order. Each chord note is channeled by the Instrument between it and the Monitor. This Instrument could also be used to affect the note's velocity, pitch and delay. Finally, the outputs are combined at the Monitor and sent to the appropriate port. This patch works best when each of the Chord Memorier's chords contains six notes since there is no convenient way to mute a middle "string" – all unused "strings" will be at the bottom of the Chord Memorizer's outlets.

In general, a setup in this style would have a Monitor leading to each virtual port of the MIDI interface. Environment patches using the various instrument object types would be constructed to control the MIDI sound device(s) on each port.

Chapter 3

Some Things Change — Some Things Stay The Same

If you are familiar with Logic's Transform window, the environment's Transformer object will look very familiar. As its name implies, the purpose of the Transformer is to change MIDI data. But, an object that simply changed everything passing through it would be of limited value, so the Transformer is also capable of selecting what data to change. What does it do with the rest? It either passes it through unchanged or throws it away.

The main difference between the Transformer object and the Transform window is that the Transformer object operates on MIDI data in real time – the data which flows through it during an environment process. The Transform window, on the other hand, makes changes to data that has already been recorded in sequences. One thing the Transform window can do that the Transformer object can't is use the position of the recorded data in its calculations. The data has no position until it's recorded and this is why the Transformer lacks position conditions and operations.

Select **Transformer** from the environment's **New** menu. A small object will appear with the default name "(Transformer)". Double-click this object and the Transformer window will appear (figure 25). This is where you tell the Transformer what to do: The Conditions section selects what data to change and the Operations section tells what to do with the selected data. At the top is a menu where you decide what to do with the unselected data. Since it's there, it provides a couple of other, convenient options. One of these options is to throw away the selected data (keeping the rest) and another is to spit the data between the Transformer's top two outlets – data matching the conditions appears at the top outlet and non-matching data appears at the second outlet. A third option is to pass the selected data unchanged as well as changed – separate choices allow the changed data to precede or follow the original data.

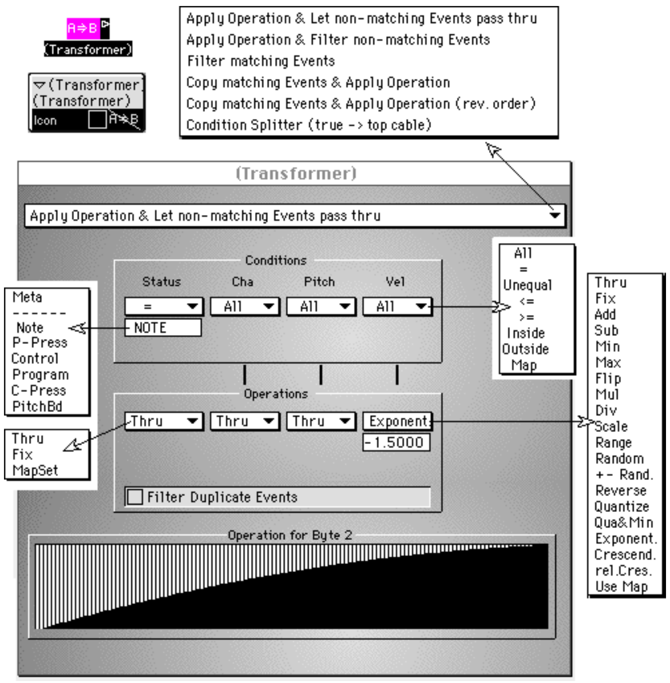
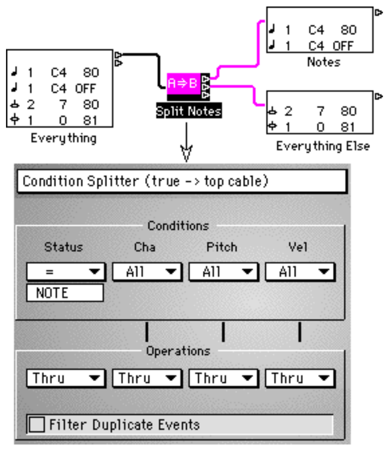


Fig. 25: Transformer window and signal path menu

One of the most basic and also most useful functions of the Transformer is to simply select data without doing anything to it, thus allowing us to split the data stream, routing some kinds of data through one process and other kinds through another. One typical split is between notes and everything else (*figure 26*). To set up a this split, select "=" from the conditions menu labeled Status and then select Note from the submenu which appears. Since we want all notes, leave the other conditions set to All. Since we don't want to make any changes, leave all the operations set to Thru. Finally, since we only want the notes, select Condition Splitter (true -> top cable) from the menu at the top of the window. Rename this Transformer to split notes so we don't have to open its window to see what it does.



```
Everything:
♩ 1   C4   80
♩ 1   C4  OFF
  2    7   80
  1    0   81

Notes:
♩ 1   C4   80
♩ 1   C4  OFF

Everything Else:
  2    7   80
  1    0   81

Condition Splitter (true -> top cable)
Conditions
Status   Cha   Pitch   Vel
=        All   All     All
NOTE
Operations
Thru     Thru  Thru    Thru
[ ] Filter Duplicate Events
```

Fig. 26: Mon->Splitter->Mon + Mon

If we set some of the operations parameters to something other than Thru, these operations will affect the notes only.

In "Condition Splitter" mode, operations apply to those MIDI events which match the conditions – non-matching events are unaffected by the operations.

You can cascade either branch of the splitting process as often as needed. Suppose for example, that you are playing a bass line with your left hand (staying below F3), playing chords with your right hand, stomping on the sustain pedal with your right foot and working the pitch bend wheel with your nose. Suppose further (if you're up to it), that you only want pitch bend applied to the bass line and only want sustain applied to the chords. *Figure 27* shows one way to do it. The everything else branch at the bottom is very important – without it, events that you're not intending to route get left out rather than sent to their intended target.

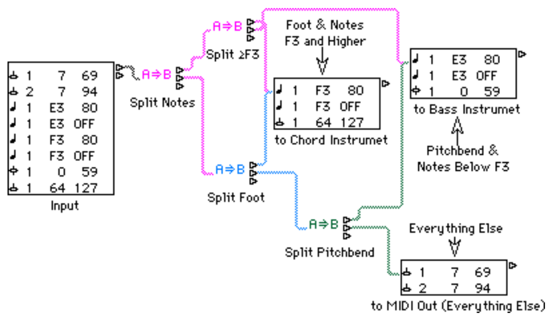


Fig. 27: Hand, Foot & Nose Controller

When you want more than one cable from either outlet of a condition splitting transformer, you must cable that outlet to some object serving as a data junction – a Monitor or neutral Transformer, for example – which allows multiple cables for the same data (*figure 28*). (Typically each new outlet from an object carries the same data but in some cases – Physical Input, Channel Splitter, Chord Memorizer, Mapped Instrument in cable split mode, Transformer in condition split mode – each outlet carries different data. Getting multiple cables from such an outlet requires a data junction.)

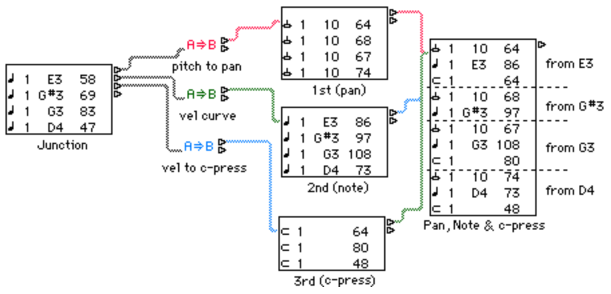


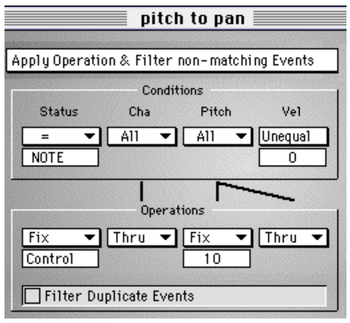
Fig. 28: Junctions and Prioritizing Data Flow

Besides connecting several cables from a single outlet, you can use a data junction to control the order of environment processes. All processing from the top outlet takes place before any processing from the second one, etc. This means that data coming out of the top outlet will be processed before any data emerges from the lower outlets.

The patch shown in *figure 28* processes each note event three times. The first (top) process converts the note to a MIDI pan event (controller #10) using the note's pitch to set the pan position. The second (middle) process scales the note's velocity thereby affecting the velocity curve of the MIDI keyboard or sequence being processed – this is the path that results in a note event at the output. The third (bottom) process converts the note to a MIDI channel pressure event using the note's velocity to set the channel pressure. The order of these three processes is important – we want pan to come first because some MIDI sound devices don't pan in real time and we want channel pressure to come last because it applies to the note(s) just sounded.

Cables from higher outlets have priority over cables from lower outlets.

Figure 29 shows the Transformer settings for the pitch to pan conversion. One thing to notice is the slanted line from the pitch condition to the velocity operation. This is how we get pitch rather than velocity to control the pan position – the line indicates that controller value (right most operation value) comes from the second condition value (pitch in the case of notes) rather than the third. Another point is that we've excluded notes with velocity 0 (i.e. note-off events). Including these would cause the pan position to jump full left at each note-off because Logic will not allow Transformers to change the right most operation value when the incoming event is a note with velocity 0. (This is a safety feature to help prevent hanging notes and you can handily defeat it by using two Transformers in series.)


pitch to pan

Apply Operation & Filter non-matching Events

Conditions

Status	Cha	Pitch	Vel
=	All	All	Unequal
NOTE			0

Operations

Fix	Thru	Fix	Thru
Control		10	

☐ Filter Duplicate Events

Fig. 29: Pitch to Pan: Data junction pitch to pan transformer

Figure 30 shows the Transformer settings for velocity scaling. The graph at the bottom shows the velocity curve that is being applied by the Transformer. The Transformer operation is set to Exponent. and the parameter is negative which results in the logarithmic curve shown.

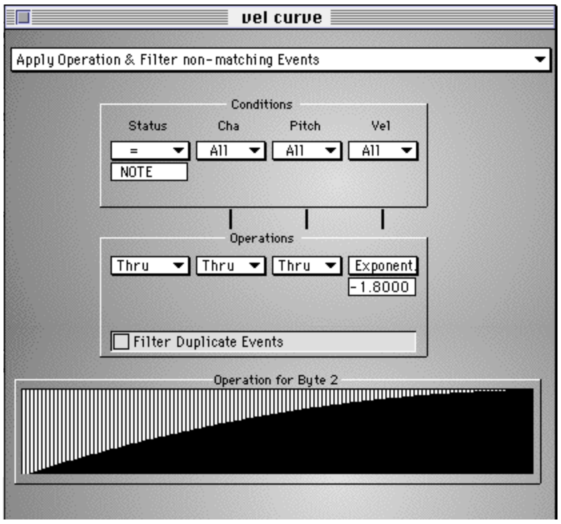
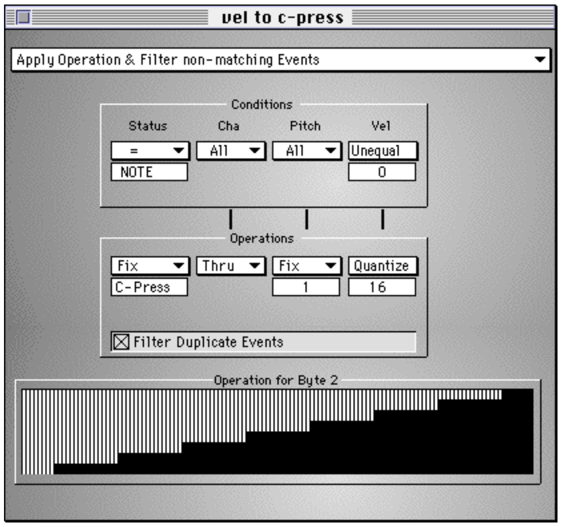


Fig. 30: Vel Curve: Data junction velocity curve tranformer

Finally, *figure 31* shows the velocity to channel pressure (often called after touch) conversion. The idea here is to break the velocity range into 8 zones by rounding the velocity to the closest multiple of 16. This is what results from the Transformers Quantize operation. Notice that the Filter Duplicate Events checkbox is checked. This prevents the same channel pressure value from being sent out multiple times in a row and is a great data saver – we'll see it used again in the data thinner example below.


vel to c-press

Apply Operation & Filter non-matching Events

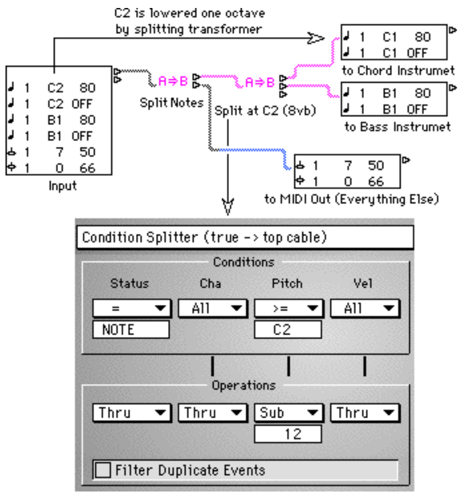
Conditions: Status = NOTE; Cha All; Pitch All; Vel Unequal 0

Operations: Fix C-Press; Thru; Fix 1; Quantize 16

☒ Filter Duplicate Events

Fig. 31: Vel to AT: Data junction velocity to after touch transformer

Figure 32 shows another application of the Transformer to split the data flow. This time, notes are sent to a splitting Transformer which divides them according to pitch and other MIDI events are passed directly to the output. This results in a keyboard split with the top zone transposed down an octave. Notice that the two notes shown in the input Monitor are C2 and B1 but that the two notes going out are C1 and B1, so that the note to the chord instrument is below the note to the bass instrument. This is the point of this patch – to avoid hand collisions while allowing pitches to overlap on a split keyboard. It is ideal for guitar / bass splits since guitar parts are scored an octave higher than they sound.



```
1  C2  80
1  C2  OFF
1  B1  80
1  B1  OFF
1  7   50
1  0   66

1  C1  80
1  C1  OFF

1  B1  80
1  B1  OFF

1  7   50
1  0   66
```

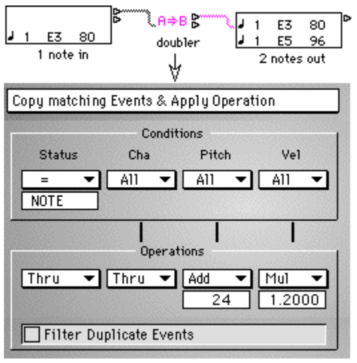
Condition Splitter (true -> top cable)

	Status	Cha	Pitch	Vel
Conditions	= NOTE	All	>= C2	All
Operations	Thru	Thru	Sub 12	Thru

☐ Filter Duplicate Events

Fig. 32: Notes split at C2 with higher zone 8vb'd

With the Transformer's Copy matching Events & Apply Operation mode, notes can serve double duty – the incoming note is passed directly to the output as well as being copied for another use. In *figure 33*, a second note is created two octaves (24 note numbers) higher. Its velocity is also multiplied by 1.2 so that it stands out.



```
1  E3  80

1  E3  80
1  E5  96
```

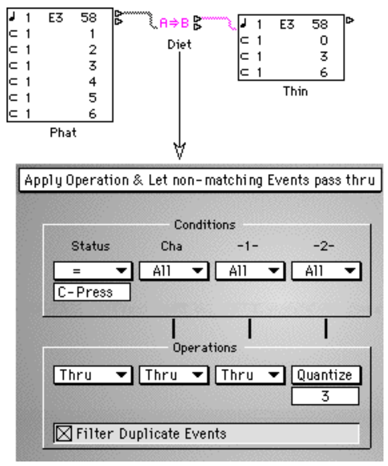
Copy matching Events & Apply Operation

	Status	Cha	Pitch	Vel
Conditions	= NOTE	All	All	All
Operations	Thru	Thru	Add 24	Mul 1.2000

☐ Filter Duplicate Events

Fig. 33: Notes are doubled 2 octaves higher (w/ velocity bump)

Figure 34 again illustrates the data thinning function of the Transformer's Quantize operation when used with Filter Duplicate Events option. In this case, MIDI channel pressure (after touch) data is quantized by the Transformer so that each incoming after touch event is rounded to the nearest multiple of 3 (i.e. the after touch resolution is reduced by 2/3rds.) With the Filter Duplicate Events option turned off, each incoming after touch message would result in an outgoing message which would likely result in a lot of duplication. With the option turned on, only changed after touch messages will be sent out most likely resulting in a significant reduction in the number of MIDI messages. For example, an incoming string 1,2,3,4,5,6 ... of after touch values would result in the output string of values 0,3,3,3,6,6 with the filter turned off but results in the output string of values 0,3,6 with the filter turned on.



Apply Operation & Let non-matching Events pass thru

Status	Cha	-1-	-2-
= C-Press	All	All	All
Thru	Thru	Thru	Quantize 3

☒ Filter Duplicate Events

Fig. 34: Quantized & Thinned after touch

The last choice on the pop-up menu for the numerical Transformer operations (i.e. the Cha, -1- and -2- operations) is called Use Map. When you select this option, a graph labeled Universal Map opens up at the bottom of the Transformer window. Below the graph are two numericals and buttons labeled Init, Reverse, Invert and Smooth. The Use Map mode allows you complete freedom in defining Transformer operations – you can map any incoming value to any outgoing value. If for example, you have a MIDI sound device which has all the standard General MIDI sounds but does not correspond to the GM preset numbers, you could use the map to convert the GM preset numbers to those which match your MIDI sound device. As another example, you could use the Transformer map to remap your MIDI keyboard although, the Mapped Instrument or Chord Memorizer is usually a better choice for this task.

Each Transformer has only one map – that's why it's labeled Universal Map – although you can use this map for more than one of the operations. You can define each of the 128 map values using the numericals below the graph – the left numerical selects what number gets mapped and the right numerical sets the map value. There are also some handy shortcuts:

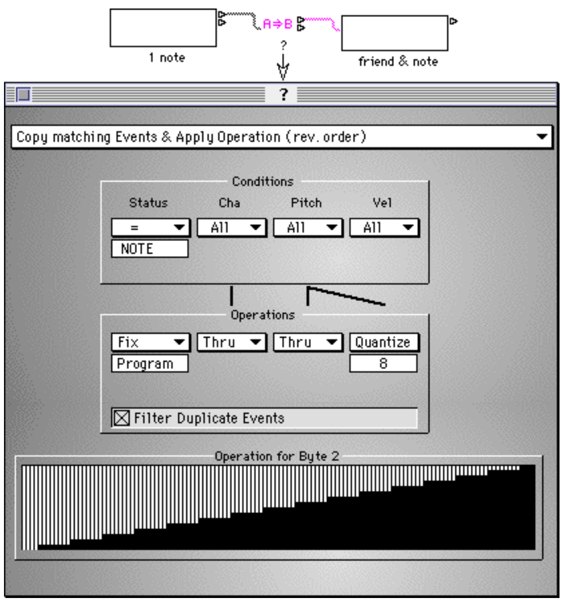
If none of the operations is in Use Map mode and none of the conditions is set to Map (see below) then any setting for the -2- operation will be reflected in the map graph although the map numericals and buttons won't appear. To fix this as the map, it is necessary to choose Use Map for one of the eligible operations (Cha, -1- or -2-) or to choose Map for one of the conditions. If you do not do this before changing the -2- operation or one of its parameters, the graph will also change. In short, any of the standard operations with any parameter setting(s) can serve as a starting point for the map. Note that when using this with one of the randomizing operations (e.g. Random or +-Rand.), once Use Map is selected the random map becomes fixed – the same input will always result in the same output.

The buttons in the map section also simplify map construction. The Init button maps each number to itself (i.e. the diagonal graph). The Reverse button reverses the position of each value (i.e. the map flips horizontally around the middle position). The Invert button tries to swap the numbers in the left numerical (the positions) with the numbers in the right numerical (the values). Since many positions can be mapped to the same value, this is not always possible so the Invert button can produce some strange results – for one thing, using the button twice will not often result in the original map. The Smooth button smoothes out large jumps in the map by adjusting adjacent values. This can be handy for producing smooth controller maps after drawing in a rough version. Repeated applications produce greater smoothing.

The condition setting labeled Map allows events to be selected or rejected based on the map value of the associated parameter. When this condition is selected, two parameter fields open up for setting a value range. When an event enters the Transformer and one of the eligible conditions (Cha, -1- or -2-) is set to Map, the event meets the condition if the mapped value of that parameter is within the range. Suppose, for example, that the Pitch condition is set to Map and that the top and bottom range boxes are both set to 0. Then note events entering the Transformer meet the condition only if their note number is mapped to 0. (Different pitches will, of course, be allowed by different maps.)

There is one other place that the word Map appears and that is in the Status operation menu. Here the choice MapSet means that the Transformer will transform incoming events which meet its conditions into a pair of events called "meta-events". (Notice that the condition and operation Status menus allow Transformers to both select and create these strange sounding events.) We'll talk about meta-events later but suffice it to say that if these events enter another Transformer they will set one of its map values. Which position's value gets set is determined by the first meta-event (a meta-event #123) and what value it's set to is determined by the second meta-event (a meta-event #122). When using the Transformer's MapSet mode, the -1- operation controls the map position and the -2- operation controls the map value.

In summary, maps may be used to set data values (via Transformer operations) or they may be used to select incoming events (via Transformer conditions). Furthermore, one Transformer can be set up to change the map values of another Transformer. Although they may seem confusing at first, maps greatly enhance the environment's power as well as simplifying patch construction. The *Environment Toolkit* contains numerous examples of this technique.



Copy matching Events & Apply Operation (rev. order)

Status	Cha	Pitch	Vel
= NOTE	All	All	All

Operations: Fix Program | Thru | Thru | Quantize 8

☒ Filter Duplicate Events

Fig. 35: Don't forget to have some fun

Stuck Notes – Things I Forgot to Say Not To Do

Every note has a beginning and an end (we hope). On your MIDI keyboard, the beginning is when you press the key and the end is when you release it. These actions, pressing and releasing, result in separate

MIDI messages – the first one is called a note-on message (NON) and the second is called a note-off message (NOF). A stuck note results when there is a note-on message which is not followed by a note-off message for the same note number and channel.

A MIDI note message (NON or NOF) contains four pieces of information: message-type (NON or NOF), channel number (0-15), note-number (0-127) and velocity (0-127). If the message-type is NON but the velocity is 0, the message is interpreted as a NOF. Logic will record real NOFs, remember their velocities and play them back as NOFs – nevertheless, Logic equates NOFs with NONs having velocity 0 and the latter form is how they are identified by the Transformer object. Setting the Transformer's Status condition to = NOTES and its Vel condition to = 0 will catch both forms of NOFs.

Notes are recorded in Logic as complete events – each note-on and its corresponding note-off is a single logical packet. It is virtually impossible to separate a note-on from its note-off once it has been recorded. (One sure way is to unplug the MIDI cable or turn off your MIDI interface during playback.) If a note enters the environment from a sequence, you can convert it to another message type, filter out the velocity =0 part and change any of its parameters but if you ever convert it back to a note, the NON and the NOF will both be there.

So, can we create stuck notes within Logic? You bet! For one thing, NONs coming in over MIDI do not have NOFs attached – if you change any of their parameters their later-arriving partners will get lost. You can also just filter out NOFs. Another thing that can result in stuck notes is converting other event-types to notes. This is true even when the original event is pre-recorded in a sequence – as long as it wasn't recorded as a note, it does not have any corresponding off event and will result in a stuck note.

If you get a stuck note here are several things you can try:

•Double-click the MIDI indicator display in the Transport.

•Press the key for the command named Full Panic.

•Play the same note again (on and off) if you can identify it.

•Turn the offending MIDI sound device off and back on.

Then there's always the elbow-to-elbow method: place your right elbow on the top note of your MIDI keyboard – place your left elbow on the bottom note – lean on your forearms. When you remove yourself from the keyboard, there may be blissful silence.

Chapter 4

Faders, Faders, Faders

All the objects we've discussed so far are designed to display, route or process MIDI data. The primary purpose of the Fader object is to create MIDI data. As the name "fader" implies, it was originally intended for creating virtual MIDI mixing consoles but its uses go far beyond that. Faders can also be used to:

•Switch MIDI data between different environment processes.

•Control Transformer condition and operation parameters.

•Program external MIDI devices with custom SysEx messages.

•Store batches of MIDI events for one-shot dumping over MIDI.

•Store snapshots of environment control settings.

•Control tempo, change screen sets and jump to markers.

Faders can also be used to process, route and display MIDI data – in a sense, they are the most versatile object in the environment. This versatility is reflected in the hierarchical Fader menu that appears when you select **Fader** from the Environment window's **New** menu (*figure 36*).

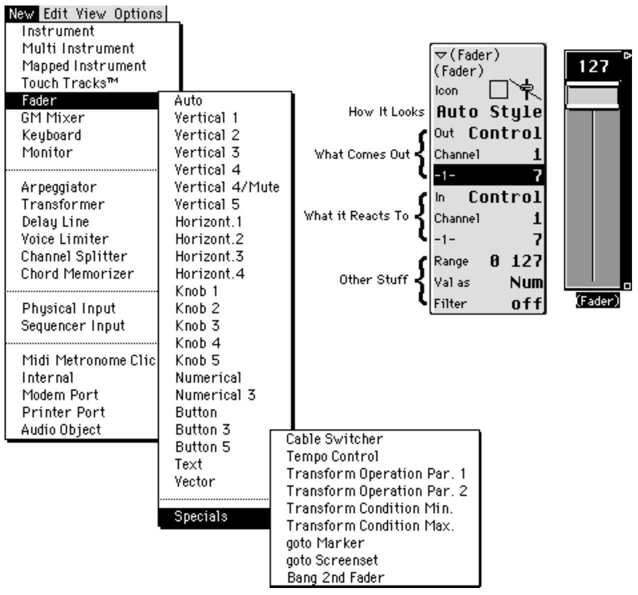
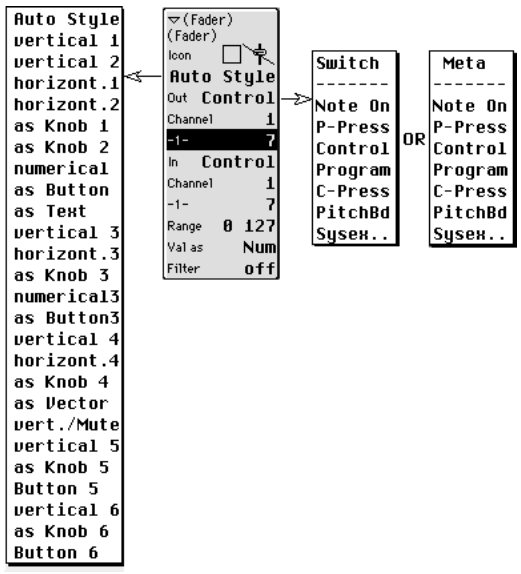


Fig. 36: "Faders" sub-menu of the New menu

There are three key pieces of information about any Fader: how it looks, what it does and what it reacts to. A Fader's appearance is completely separate from what it does. You decide a Fader's appearance either when you select it from the Fader sub-menu or by using the style menu in the Fader's parameter box (*figure* 37).

What a Fader does is determined by its Out definition which is set in its parameter box (*figure* 37). What it reacts to is determined by its In definition, also set in the parameter box. The Out definition has three parameters labeled Out, Channel and -1-. Out is where you choose the type of MIDI event the Fader sends out. Channel sets the MIDI channel of the event. For controller events, the -1- parameter determines the controller number. For other event types it has other meanings but we'll get to that later.



```
▽(Fader)
(Fader)
Icon
Auto Style
Out     Control
Channel       1
-1-           7
In      Control
Channel       1
-1-           7
Range     0 127
Val as      Num
Filter      off
```

Fig. 37: Style and I/O definition menus

The simplest way to use a Fader is to connect its outlet to a port and change its value with the mouse. When you create a new Fader, its Out definition is set to control MIDI volume (controller #7) on channel 1 and its style is set to Auto Style. If you cable it to any port object and have a MIDI sound device listening on channel 1, this Fader will control its volume. If you make 15 copies of this Fader; set their Out definition channels to 2 thru 16 and cable each of them to the same port, you have a simple MIDI mixer. But don't do it just now – we're going to build a better one below and learn a few tricks in the process.

The are four basic styles of Faders: sliders, buttons, text and vector. Sliders come in several varieties – vertical, horizontal, circular (i.e. knobs) and numerical. The thing they all have in common is that you can scroll through their range of values using the mouse. Some sliders display their value numerically and for those, you can type in their value by double-clicking on the number. (Numericals have only this type of display but they can still be scrolled with the mouse like other sliders.)

For any Fader with a numerical display, you can scroll by single steps if you grab the numerical rather than the slider handle.

You set a Fader's range of values in the Range section of its parameter box. The maximum range is 0 thru 127 which is the MIDI data value range. For SysEx Faders which we'll discuss later, this range is -128 thru 9999.

You can also set up one Fader to control another Fader's range limits. (See the section on meta-events below.)

Unlike sliders, buttons have only two possible values and you click them to change between the two. The two possible values are the minimum and maximum of the value range.

Text-style Faders are like sliders in allowing you to scroll through their range of values, but unlike sliders, each value can be given its own name. Also, text Faders can be configured to pop up like menus rather than to be scrolled with the mouse. You enter a text Fader's names by double-clicking on it to open a window with 128 name positions. If a text Fader's range contains fewer than 128 values, only that number of name-slots is available and regardless of the range minimum, the names always start in the top-left slot. Names can be copied & pasted to and from text Faders and it is often easiest to edit them in a simple text editor. Also, names from Multi-Instruments can be pasted directly into text Faders and vice-versa.

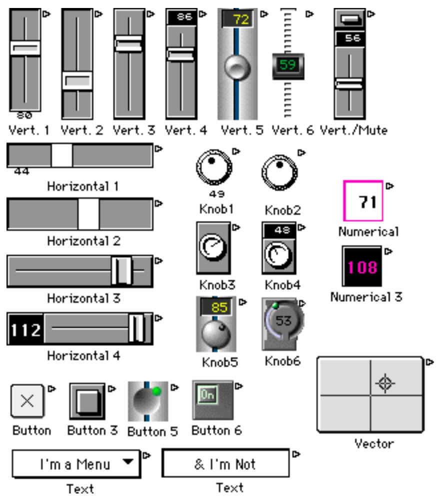


Fig. 38: All Fader Styles illustrated

Vector style Faders are like two-dimensional sliders – horizontal movement changes one control value and vertical movement changes another. For vector Faders, the In and Out definitions are replaced by Vert and Horz definitions, respectively. The channel and message type for vertical movement is set by the Vert definition. For horizontal movement, it is set by the Horz definition. Each mousing of the vector Fader results in two values, even if one of them hasn't changed. You can thin out these repeated values by inserting a neutral transformer after the vector fader and checking its Filter Duplicate Values checkbox. Incoming data (e.g. from another Fader) which matches one of the vector's definitions will affect only the corresponding dimension and will result in output of only that dimension's values.

If the Vert and Horz definitions of a vector Fader are the same, the vector Fader becomes a four channel Fader. Each mousing of the vector causes events to be output for four consecutive channels starting with the channel set in the Vert and Horz definitions. This is great for controlling four mixer channels or two channels and their pans.

The values output for each of the four channels are (roughly) a measure of the distance of the mouse from each of the four corners of the vector Faders window. If for example, the channels are 1 thru 4 then numbering the corners as in *figure 39* then the closer the mouse is to a channel's corner, the higher the value.

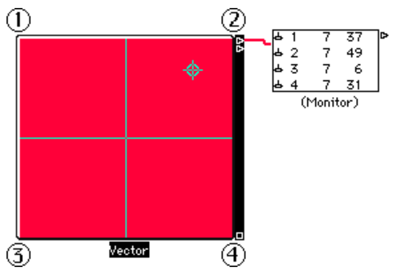
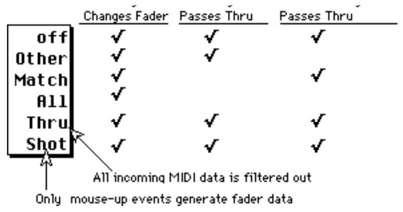


Fig. 39: Vector fader with numbered corners (closer is higher)

At the bottom of the Fader parameter box is a parameter labeled Filter (*figure 40*). This acts much like the Transformer's conditions section – it determines what incoming MIDI events the Fader reacts to and/or allows to pass through. The choices indicate what items are filtered (i.e. blocked) by the Fader with reference to the Fader's In definition. In the case of Match or All there is no Fader output for incoming data matching the In definition but the Fader's display will still be updated. The Thru setting blocks all data coming from MIDI input to Logic (see *The Absolutely, Guaranteed Simplest Environment Patch in the World* below) but allows all data recorded in Logic sequences. The Shot setting is not really a filter-setting – with this setting mousing the Fader results in only the final value's being sent (i.e. the value when the mouse is released).



	Changes Fader	Passes Thru	Passes Thru
off	✓	✓	✓
Other	✓	✓	
Match	✓		✓
All	✓		
Thru	✓	✓	✓
Shot	✓	✓	✓

All incoming MIDI data is filtered out
Only mouse-up events generate fader data

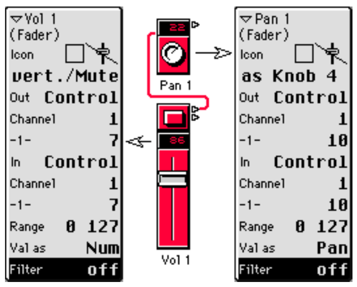
Fig. 40: Fader Filter Menu (Annotated)

4.1 Building a Simple MIDI Mixer

Since this is the intended and most straight forward application of Faders, let's see what it involves. One thing to recognize is that a MIDI mixer is a different animal than the mixing consoles you're probably used to. Not only is it different because it involves MIDI data rather than audio – it is different in principle, in that it is not affecting something passing through it but rather is sending out MIDI information of its own. The information being sent out controls MIDI sound devices by changing the volume and pan parameters of their presets. The important thing to keep in mind is that this is done on a channel-by-channel basis – any MIDI sound device listening on a given channel and port will have its volume and pan controlled by any MIDI mixer information sent on that channel.

As we'll see below, it is possible and often desirable to put the MIDI mixer in the "signal path" by having the MIDI note events pass through it. The *Environment Toolkit* contains a complex "signal path" mixer which among other things, features trim controls and an effects bus.

Mixers of any kind look repetitive because they are. For our simple MIDI mixer we start with a "module" consisting of a volume and a pan control then repeat this module for each desired channel. *Figure 41* shows how the basic module and how it is wired.



▽Vol 1 (Fader)	
Icon	
vert./Mute	
Out	Control
Channel	1
-1-	7
In	Control
Channel	1
-1-	7
Range	0 127
Val as	Num
Filter	off

▽Pan 1 (Fader)	
Icon	
as Knob 4	
Out	Control
Channel	1
-1-	10
In	Control
Channel	1
-1-	10
Range	0 127
Val as	Pan
Filter	off

Fig. 41: How the basic mixer module is wired

For the volume Fader we've chosen the vertical-mute style and for the pan Fader we've chosen a knob. There is a single cable from the volume Fader to the pan Fader. To install this module, we would connect the output of the pan knob to the port serving the MIDI sound device whose channel it is meant to control.

Take a look at the parameter boxes for each control. Notice that their In and Out definitions match so that if cabled into the data path, they would display any incoming settings. We will use this feature later.

The pan knob's controller number is set to 10 and the volume slider's controller number is set to 7 – these are the standard MIDI pan and volume controller numbers.

The Fader's Val as parameter (*figure 42*) determines how the numerical display will represent the current Fader value. For the pan knob this is set to Pan which means the value range (0 thru 127) will be displayed as -64 thru 63 representing far left to far right panning. For the volume knob the Val as parameter is the default setting of Num – the actual data value is displayed. All the other choices except Hz divide the value range into equal steps with displays appropriate to various other Fader functions. Hz grows exponentially over the value range.

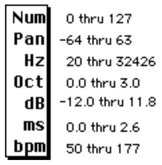


Fig. 42: The Val as menu - annotated

To create our simple MIDI mixer we need first to copy this module for as many channels as we wish to include. Next we need to change the In and Out definition channels of each Fader in each module and to change the name of each Fader in each module. Finally we need to create a MIDI signal path through each of the volume and pan Faders. Logic provides shortcuts for each of these steps which we'll illustrate by creating a sixteen channel mixer.

First, disconnect the cable from the volume Fader to the pan Fader in the single module we created above – we'll recable the 16 channel mixer in a different way. Next, duplicate the volume/pan module 15 times for a total of 16 modules (one for each MIDI channel). The easiest way to copy the module is to select both Faders then while holding down the ⌥ key, click on the name section of one of the Faders and drag. Do this to create 2 modules from 1, select all 4 Faders and do it again to create 4 modules from 2, select all 8 Faders and do it again to create 8 modules from 4 and finally, select all 16 Faders and do it again to create 16 modules from 8. You should now have 16 modules looking like *figure 43*.

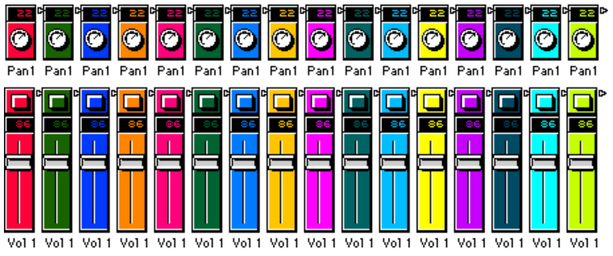


Fig. 43: 16 identical mixer modules

The next step is to select all the volume Faders. The top of the parameter box will now say Multiple Sel. Click on this and an editing box will pop open. Replace Vol 1 by #1 in this box and press the return key. The 16 volume Faders will now be named #1 thru #16. Open the editing box for the 16 pan knobs in the same way but this time simply delete Pan 1 so that the knobs have no name.

The next step is to change the channels. Select the Fader named Vol 1 and choose **Copy** from the **Edit** menu. Now select all 16 volume Faders; open the Options menu and choose **Definition, channel increment** from the **Apply Buffer Template to** sub-menu (*figure 44*). (Copying the selected Fader is what created the buffer template – it has the same effect as choosing **Define Template** from the sub-menu.) Repeat this process for the pan Faders and then look at each Fader's parameter box to satisfy yourself that the modules are correctly channeled.

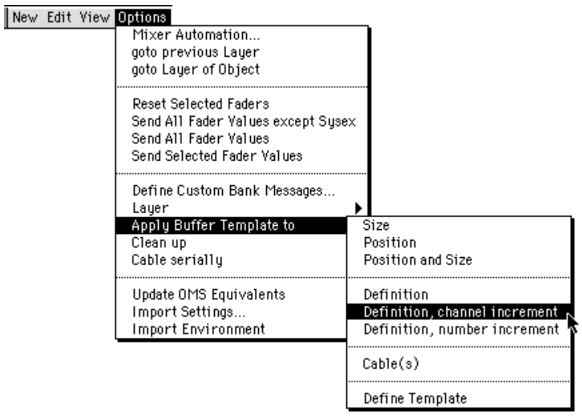


Fig. 44: Apply buffer template sub-menu

The final step is to create a chain of cables from the first volume fader to the last pan fader. When building a complex mixer you may want to keep each module separate in order to use different ports or apply different processing for each channel, but for a simple mixer with all outputs going to the same port, cabling all Faders in series is the easiest solution. To do this, you don't need to make 32 separate cable connections – Logic provides a shortcut. Select all the volume Faders and from the **Options** menu select **Cable serially**. This will cable each volume Fader to the one to its right, leaving the rightmost one uncabled. Cable this Fader to the leftmost pan knob (the one for channel 1). Next select all the pan knobs and repeat the process. Finally, connect a cable from the rightmost pan knob to the desired port (or to a Monitor if you want the mixer to be importable) and except for a little housecleaning, you're done (*figure 45*).

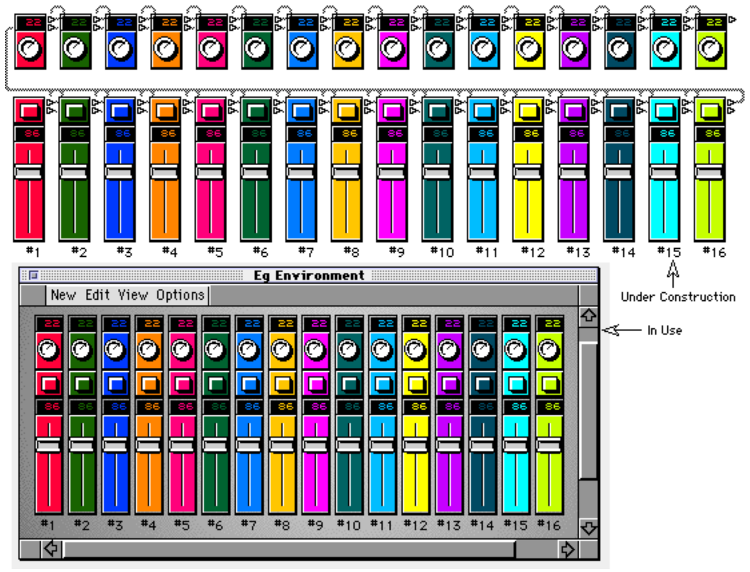
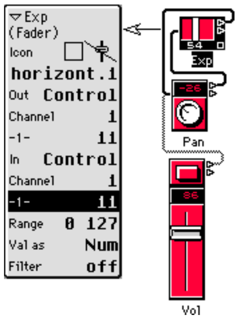


Fig. 45: finished 16 channel mixer w/ protected view as insert

In practice (especially with larger mixers) it is confusing to see the cables and have the modules so far apart. To get the controls as close together as possible, first select **Cables** from the **View** menu until it has no "•" beside it (i.e. the cables and outlets disappear). Next, select the 16 pan knobs and choose **Align Objects** from the **Clean Up** sub-menu of the **Options** menu. Now do the same for the volume Faders. Finally from the **View** menu turn **Protect Cabling/Positions** on ("•") and turn **Parameters** off (no "•"). The Environment window should now look like the insert in *figure 45*.

Any serious audio mixer has more than pan and volume with mute. Some of the things you find there have no real MIDI mixer analogy (effects send/return, trim and monitor are examples) because a MIDI mixer is not processing something passing through it. (The "signal path" mixer in the *Environment Toolkit* does feature trim controls and effects busses.) On the other hand, controls like EQ may have a counterpart among the MIDI sound device preset parameters. There's nothing to prevent you from adding any controllers you like to the volume/pan chain but bear in mind that if you change the MIDI sound device listening to a controller's channel, the result may also change – pan and volume are (nearly) universal but the effect of most other MIDI controllers is "device-specific". Also remember that since you're building the mixer yourself, there is no reason all channel modules have to be the same. A pan knob is usually useless on a drum module, for example, so why include it? Different appearing modules are also more easily recognized.

Figure 46 shows the simplest way to add another controller Fader which we've named Exp. The circular connection from Pan to Exp and back to Pan looks as if it will cause MIDI feedback but Logic prevents this. The result is that moving Exp sends its MIDI messages through Pan to the output while events entering the module at Vol pass through Pan to the output and to Exp for display updating. You can add as many new controls as you want in this way and in fact, you could connect Vol directly to the output and then cable Pan this way, too.

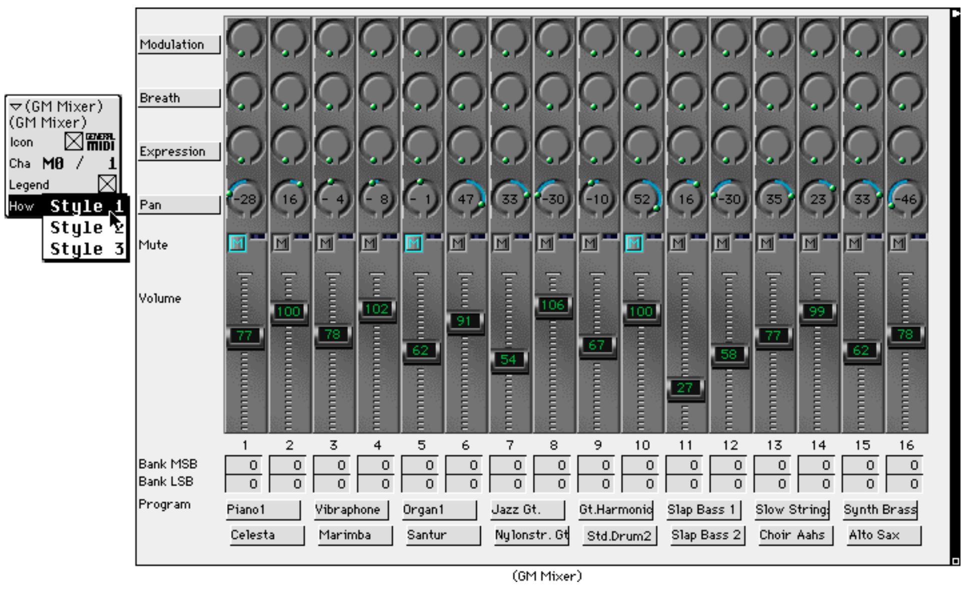


```
▽Exp
(Fader)
Icon
horizont.1
Out      Control
Channel        1
-1-           11
In       Control
Channel        1
-1-           11
Range      0 127
Val as       Num
Filter       off
```

Fig. 46: Adding a knob to a mixer

4.2 The GM Mixer

Logic contains a ready made, 16 channel mixer called **GM Mixer** on the environment's **New** menu. The GM Mixer (*figure 47*) contains volume/ mute sliders, bank and preset numericals and four additional controller knobs for each of the 16 MIDI channels. The four controller knobs can be assigned to any MIDI controller – by default they are assigned to pan (#10), portamento (#5), chorus depth (#91) and reverb (#93). The GM Mixer supports Roland's GS MIDI standard and Yamaha's XG MIDI standard.



(GM Mixer)
(GM Mixer)
Icon
Cha M0 / 1
Legend
How Style 1
Style 2
Style 3

Fig. 47: Illustration of GM Mixer

The only GM Mixer parameters are Legend which determines whether the fader names appear on the left (when off the mixer is smaller), How which determines the style of the mixer's Faders and knobs and Cha which determines the port for the mixer's default cabling (just as with Instruments) and channel. The channel setting also determines the lowest (leftmost) mixer channel and therefore controls the size of the mixer since it always goes up to channel 16. Place a Transformer after the mixer to change the channel offset – for example, set the GM Mixer's channel to 11 then set the Transformer to subtract 10 from the channel to get a mixer for channels 1 thru 6. (Note that if there is data coming into the GM Mixer, you'll need a Transformer before the mixer to add 10 to the channel in order for the input and output channels to correspond.)

There are three disadvantages to the GM Mixer: you can not take snapshots of its Fader settings (see *Automating the Mix* below); its program selectors always use the GM preset names and it can't be banged (see the section *Q. What's a Meta?* below) . A major advantage to the GM Mixer is that its mute button is not toggled off by moving the corresponding slider (as is the case with Logic's Vert./Mute style Fader). The GM Mixer does put out multiple volume 0 events when the slider is moved but a Transformer after the GM Mixer with its Filter Duplicate

Events option turned on can be used to suppress these duplicate volume events.

4.3 Automating the Mix

The most obvious way to use a MIDI mixer is to mouse the various sliders and knobs. This will send the corresponding controller values to whatever output port(s) the mixer is cabled to. Ya, ok ... next.

A more interesting use is to record the Fader movements. Fortunately, you don't have to do anything more in order for this to work. Fader movements are automatically sent to the currently selected track and if recording is going on, they will be captured. Of course, they will then be played back over the instrument for that track and subject to any processing this may involve. In particular, if an Instrument is used it will typically re-channelize everything. So, recording Fader movements on an Instrument track is best suited for volume & pan changes on a single channel – the Instrument's channel.

For multi-channel mixing, use a generic track – one whose instrument is the port itself or an Instrument whose channel is set to All, for example. It may seem like this will result in duplicate controller events being sent over MIDI but this does not happen. While the Fader movements are being recorded, they are not being passed through to the track's instrument. On the other hand, while they are being played back, they are not being sent from the mixer itself. So, in neither case are there duplicate events.

If you want the MIDI mixer to display the results of an automated mix, you need to create a new environment object and cable it to the beginning of the volume/pan Fader chain. In the Logic default song, an Instrument named Playback whose channel is set to All is used. Other good candidates for this object are a Channel Splitter, Transformer or Monitor. Again, because information is not passed through during recording, you can record and playback on a track which uses this Playback instrument.

Logic's transport does not need to be running to record Fader movements. In Record-Pause mode, moving a Fader will cause a snapshot of its position to be recorded on the selected track.

4.4 A Mixer for Each Port

In the *What's It Good For* section we constructed a two port setup. One approach to mixing consoles is to create a separate mixer for each port, customized for the specific MIDI sound devices on that port. To add the 16 channel mixer above or a GM Mixer to the setup for the first port in that example simply cable the Multi-Instrument into the #1 volume slider (or GM Mixer) and cable the last pan knob (or GM Mixer outlet) into the Channel Splitter. One advantage to placing the mixer after the Multi-Instrument is that any active volume and pan sub-instrument parameters will be displayed by the mixer. If you want a separate "mix" track, you can use the Multi-Instrument for the track instrument instead of creating a separate "playback" Instrument.

Figure 48 uses the same approach to add a GM Mixer to the Chord Memorizer guitar setup we constructed for second port in our example. Notice that the GM Mixer reflects the individual "String" Instrument's parameter settings. To use this setup, select the Chord Memorizer as a track's instrument. Choose **Used Instruments MIDI Settings** from the **Send to MIDI** sub menu of the Arrange window's **Options** menu to send all the "String" Instrument parameters thru the GM Mixer to the MIDI sound device connected to port #2.

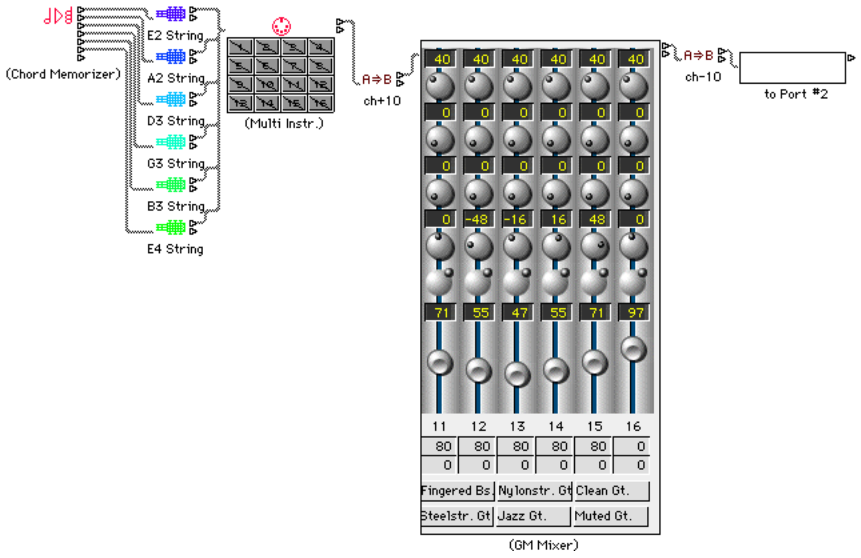


Fig. 48: Adding GM Mixer to Chord Memo basic setup

4.5 A Signal Path Mixer with Real Muting

In the example above, the GM Mixer is in the MIDI signal path – all MIDI note events pass through the GM Mixer on their way to their port. But, this is merely a convenience to get the GM Mixer to show incoming (real time or recorded) mixer events. The GM Mixer doesn't actually affect the notes passing through it. Let's call this an "in line" mixer to distinguish it from a mixer environment which works by affecting the MIDI note events passing through it. We'll call a mixer that does affect MIDI note events a "signal path" mixer.

One thing that a signal path mixer will share with any other MIDI mixer is that it will still control MIDI volume and pan position by sending MIDI controller events. Beyond that what can we do? The parameters of a MIDI note event are its pitch (note number) and velocity so one thing we can do is affect these parameters. In addition, we can control a note's destination and create alternate destination "busses" for additional environment processing. The *Environment Toolkit* features a mixer environment which does all these things and also includes a master fader, but for now let's create a simple signal path mixer with a different kind of mute button.

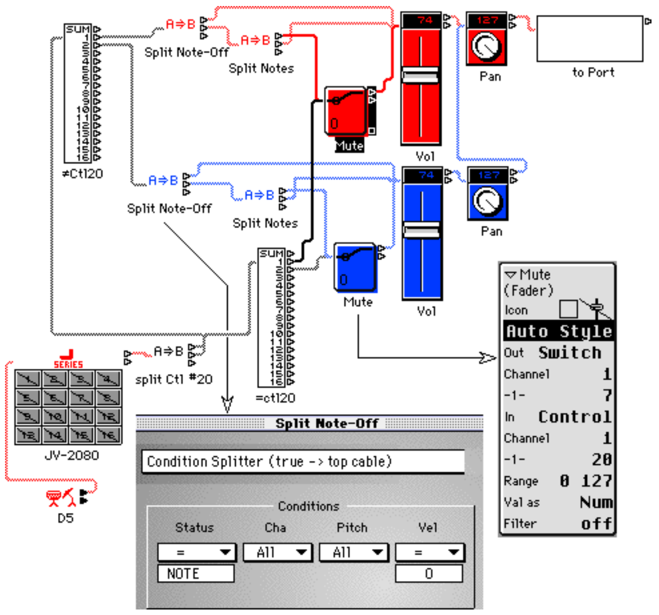
As with the 16-channel mixer we constructed in the *Building a Simple Mixer* section, we will start with a single channel module then duplicate it until we have all 16 channels. This time, the module will have four elements – volume and pan Faders, a muting switch and a Transformer to split notes from other MIDI event types. Since we're constructing a separate mute button, the volume Faders don't need to have the mute button on top so we've used a different style of Fader. For the mute button we'll use a special kind of Fader called a Cable Switcher.

At the bottom of the **Faders** sub menu of the Environment window's **New** menu, there's another sub-menu labeled **Specials**. The first choice from this sub menu is **Cable Switcher**. A Cable Switcher has In and Out definitions like any other Fader but the Out definition is irrelevant because rather than creating MIDI events, the Cable Switcher simply routes incoming MIDI events to one of its outlets. The number of outlets a Cable Switcher has is determined by the number of cables that are drawn from it – each outlet can have only one cable leaving it and when an outlet is cabled, a new outlet appears below it. The maximum number of outlets is 128, numbered from 0 to 127 corresponding to the MIDI data range. Clicking the Cable Switcher steps through its outlets, but outlets can also be selected by number using MIDI events matching the Cable Switcher's In definition. (These events are *not* passed through the Cable Switcher so if you want a Cable Switcher to route everything you must give up MIDI control and set its In definition to `---------`.)

A Cable Switcher can have any style and for our mute button we've used the Button 3 style. In this case the out position (dark center) routes data to the first outlet (position #0) and the in position (white center) routes data to the second outlet (position #1).

We want the mute button block MIDI note-on events but let other MIDI events pass thru. The reason for this is twofold: we want incoming volume and pan events to be reflected by the mixer and sent to the MIDI port even when the channel is muted and we don't want to leave hung notes when a channel is muted while a note is being held. In order to accomplish this we've added two splitting Transformers – one to split note-off events from everything else and the next to split note-on events from the remaining MIDI data. The reason two Transformers are needed is that, as a safety feature to prevent hung notes, a Transformer will not select just note-on events.

Figure 49 shows the cabling for a module of the signal path mixer. The top outlet of the note-off splitting Transformer (the cable that carries note-off events) is cabled directly into the signal path through the volume Fader. The second outlet from this Transformer is cabled to the note-on splitting Transformer. The top outlet of the note-on splitting Transformer (the cable that carries the note-on events) is cabled to the mute button and the second outlet is cabled to the volume Fader. The volume Fader is cabled to the pan knob. Once 16 modules have been created, the pan knobs should be cabled in series and the channels should be incremented just as was done with the simple 16-channel mixer above. There's no need to cable the volume Faders in series because each is "fed" by its own mute button. (The mute buttons are not connected in series because otherwise, muting a channel would also mute all higher numbered channels – not the desired outcome.)



Split Note-Off

Condition Splitter (true -> top cable)

Conditions

Status	Cha	Pitch	Vel
=	All	All	=
NOTE			0

Mute (Fader)

Icon	
Auto Style	
Out	Switch
Channel	1
-1-	7
In	Control
Channel	1
-1-	20
Range	0 127
Val as	Num
Filter	off

Fig. 49: A single module of the signal path mixer

In *figure 50* we've used the signal path mixer in conjunction with the Multi-Instrument / Channel Splitter from the basic setup we constructed in the *What's It Good For* section. Instead of the Channel Splitter's SUM outlet going directly to the Monitor, to Port #2, each channel is cabled separately to the corresponding splitting Transformer. Then we've cabled the last pan knob to the output Monitor.

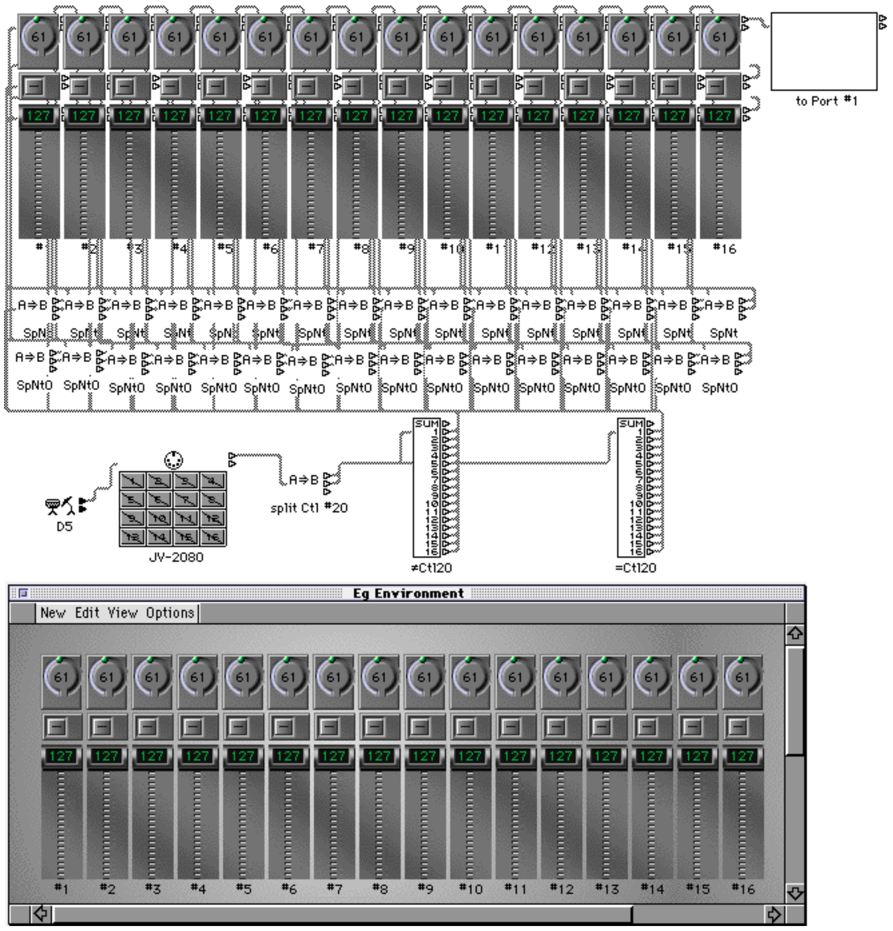


Fig. 50: Simple signal path mixer w/ mute buttons

We've added one other feature – programmable mutes – by splitting off controller #20 events from everything else and sending them through a separate Channel Splitter to the mute buttons. (The mute buttons' In definitions have been set to control #20 on channels 1 to 16.) This allows mute events to be entered manually into sequences or input from an external MIDI controller in real time.

4.6 Q: What's a Meta?

A: A river in northeast Colombia flowing partially along the border with Venezuela.

B: Two positions in the benzene ring separated by one carbon atom.

C: An object of a higher logical type.

D: All of the above.

E: Something else entirely.

Faders, except for the Cable Switcher, send out events. Mostly these are MIDI messages but occasionally they are messages to Logic itself and not intended for MIDI consumption. These are called "meta-events".

Meta-events don't have to come from Faders and they are not just used in the environment. Many of the special notation symbols in a score are meta-events and if you open a scored sequence's Event list, chances are you'll see some meta-events in there. In fact, you can create meta-events directly in the Event editor by holding down the ⌘ key and clicking the full message button – the one with the 0's and 1's on it. If you now scroll the value in the Num column you will see the meta-event type names as they pass. Here are the relevant ones for controlling Logic:

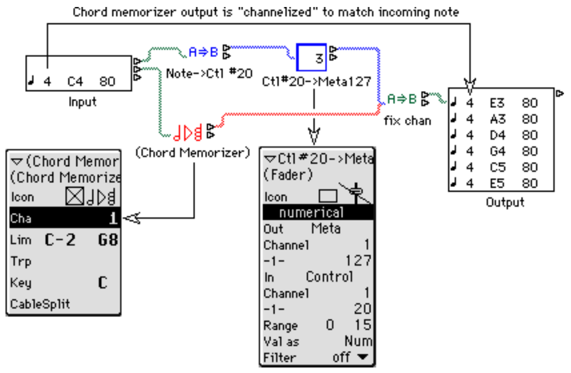
Meta-event Number	What it Does
127	*Sets top Transformer operation parameter.
126	*Sets bottom Transformer operation parameter if there is one.
125	*Sets top Transformer condition parameter.
124	*Sets bottom Transformer condition parameter if there is one.
123	*Sets Transformer's map position.
122	*Sets Transformer's map value (for current position).
100	Sets tempo to value+50 (temporary until stop or cycle jump).
99	Bang - causes Faders to resend their current value.
98	Silent - sets the Fader value without sending it.
97	Sets Fader range maximum.
96	Sets Fader range minimum.
52	Stops Transport.

51	Sets locators by marker number and jumps to left locator.
50	Selects song.
49	Selects screen set.
47	†Sends a single MIDI byte. (Use several to roll-yer-own.)

* Events must get to the Transformer
†Can not be used as a meta-fader's "-1-" parameter

Meta-events can be entered directly into sequences in the Event editor, they can be generated by meta-Faders and they can be entered directly into SysEx buttons (see below). As we will see time and again, the ones that control Transformer parameters (122 - 127) are extremely powerful. They are the ones that allow the environment to speak in a higher logical type.

Figure 51 shows a simple example of the usefulness of this combination. Recall that the Chord Memorizer is a "channelized" object – all output is channelized according to its Cha parameter setting. One way to avoid having to reset this parameter is to follow the Chord Memorizer with a Transformer to fix the channel as desired. In order to have the channel "follow" the input, notes are first converted to controller events (control #20 in this case) which match the meta-Fader's In definition. The meta-Fader in turn, sets the Transformer's channel operation Fix parameter. So, incoming notes first set the Fix parameter then are sent to the Chord Memorizer to generate the desired chords.



4 E3 80
4 A3 80
4 D4 80
4 G4 80
4 C5 80
4 E5 80

▽(Chord Memor
(Chord Memorize
Icon
Cha 1
Lim C-2 G8
Trp
Key C
CableSplit

▽Ctl#20->Meta
(Fader)
Icon
numerical
Out Meta
Channel 1
-1- 127
In Control
Channel 1
-1- 20
Range 0 15
Val as Num
Filter off

Fig. 51: Channelizing the CM with a meta-fader/ transformer combo

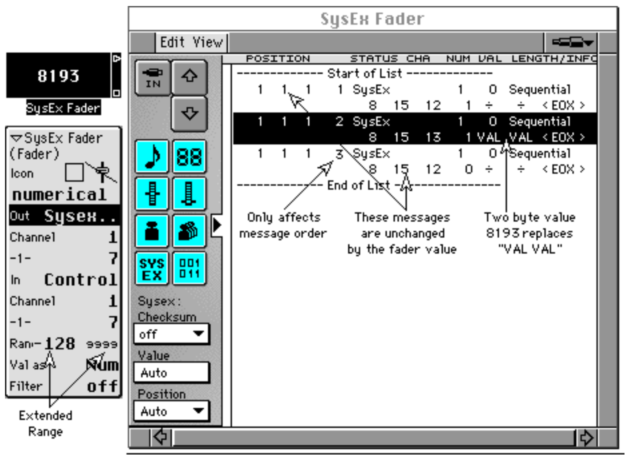
Another useful meta-event is the bang event (#99). This event causes Faders to resend their current value and in this sense it functions as a kind of Fader memory. (Recall that events entering a Fader normally either change its value or pass through unchanged.) One use for bang Faders is as a setup button – connect a button style bang Fader to all Faders in a mixer. Here are some things to remember about bang events:

- Bang events must use channel 1 – the channel of the target Fader is irrelevant.
- Bang events pass through Transformer – you can use a Transformer as a junction.
- Bang event values of 123 and 127 cause the target Fader value to increment.
- Bang event values of 121 and 125 cause the target Fader value to decrement.
- Bang event values of 121 & 123 cause Fader "rollover" while values 126 & 127 do not.
- Bang event value of 1 simply causes the current Fader value to be resent.
- •The bang events listed above do not "bang thru" the target faders to other Faders in the chain.
- •Subtracting 1 from the above bang values produces bang events which DO "bang thru".

4.7 A Little More About SysEx Faders

As the name implies, SysEx Faders were originally intended for sending MIDI system exclusive messages – the messages usually used to set MIDI sound device preset parameters not accessible via standard MIDI controller messages. In brief, "system exclusive" means exclusively used by the system – in this case the system is the MIDI sound device. The first part of a SysEx message (after the status byte which identifies it as a SysEx message) identifies the system to which it is exclusive. This usually means a MIDI manufacturer, a family of that manufacturer's products and a specific model within the family. Everything that comes after this is peculiar to the system involved – there are no standards – until the end when another status byte announces the end of the message.

A SysEx Fader works from a list of MIDI messages some or none of which may be SysEx messages. When the Fader is moved (or when a message matching its In definition arrives), the entire list of messages is sent out. When you open the message list window (*figure 52*) by double-clicking the word SysEx in the Out definition, you can edit the messages just as in Logic's list editor. The only difference is that the position has no effect other than to order elements within the list. When you close the window, Logic remembers which events in the list were selected and which were not. For the selected events, the Fader or incoming value is substituted for the value displayed in the list. The unselected events are sent as written.



```
SysEx Fader

POSITION       STATUS  CHA    NUM  VAL  LENGTH/INFO
------------- Start of List -------------
  1   1   1    1  SysEx          1    0   Sequential
                     8   15   12    1  ÷    ÷   < EOX >
  1   1   1    2  SysEx          1    0   Sequential
                     8   15   13    1 VAL  VAL  < EOX >
  1   1   1    3  SysEx          1    0   Sequential
                     8   15   12    0  ÷    ÷   < EOX >
------------- End of List -------------
```

```
8193
SysEx Fader
▽SysEx Fader
(Fader)
Icon
numerical
Out      Sysex..
Channel        1
-1-            7
In       Control
Channel        1
-1-            7
Ran- 128    9999
Val as       Num
Filter       off
```

Sysex:
Checksum: off
Value: Auto
Position: Auto

Edit
View
IN
Only affects message order
These messages are unchanged by the fader value
Two byte value 8193 replaces "VAL VAL"
Extended Range

Fig. 52: SysEx fader list window

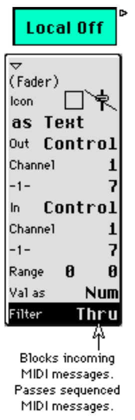
For non-SysEx messages, the VAL position is what is replaced. For SysEx messages the situation is a little more complicated. You can choose what position is replaced and you can even choose more than one position. Also you can specify the form of the replacement. Finally, you can call for a checksum. All of these choices require knowledge of how the receiving system works – knowledge which is anywhere from hard to impossible to glean. Sometimes, there is an easier way.

The SysEx window has an IN button which allows it to listen to incoming SysEx messages. If you click this button, then send a SysEx message from your MIDI sound device, it will appear in the window. (One message at a time, please.) The message shown here, for example, was sent by a JV-1080 when its Output Assign parameter was changed on the front panel . Pressing the button now toggles that value between 0 and 127 on the JV-1080.

Whether or not you get into SysEx, don't forget about SysEx Faders. They're great buckets for lists of MIDI messages and can be used to setup both external MIDI sound devices and internal environment patches.

4.8 The Absolutely, Guaranteed Simplest Environment Patch in the World

Figure 53 is it – a text Fader with no Out or In definition. This Fader does absolutely nothing when you mouse it. Whatever comes into it from Logic (i.e. from a sequence or other environment patch) passes through unaltered.



```
Local Off
(Fader)
Icon
as Text
Out   Control
Channel     1
-1-         7
In    Control
Channel     1
-1-         7
Range   0   0
Val as    Num
Filter   Thru
```

Blocks incoming MIDI messages. Passes sequenced MIDI messages.

Fig. 53: Local Off - fader filter "thru" mode

However, whatever comes into it from MIDI (i.e. as a track's instrument or patched from Physical Input) is blocked. This is the meaning of the filter parameter Thru. The point of defining an object like this is to prevent MIDI doubling and feedback.

MIDI doubling occurs when a MIDI controller sends data directly to another MIDI device as well as into Logic. If Logic passes the data through, it will reach the receiving MIDI device twice. MIDI feedback occurs when the receiving MIDI device sends the MIDI data it receives back thru its MIDI out port.

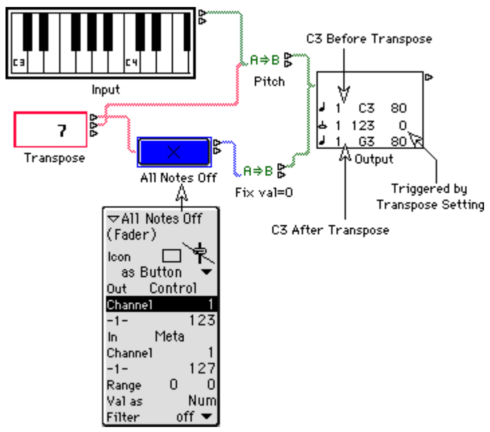
Normally these happen when the controller is internal to the device and they can usually be prevented by changing the device's "Local On" or "MIDI thru" settings. But, some older MIDI devices don't have these options.

In the **MIDI Options** section of Logic's **Song Settings** you can designate one item on the Instrument menu for which to disable the MIDI thru function. Putting this Local Off Fader after the Instrument will accomplish the same thing. And, of course, you can have more than one of them.

4.9 The Fader As Transformer

It is probably obvious by now that one of the things a Fader does is transform incoming data matching its In definition to data of the type and channel specified by its Out definition. This is similar to setting a Transformer's Status and Cha conditions to match the Fader's In definition and setting its Status and Cha operations to match the Fader's Out definition. So, why not just use a Transformer? There are two reasons: you can use a Fader as a screen object to generate data and you can't transform meta-events 122 thru 127 with a Transformer.

Recall that meta-events 122 thru 127 are used to set Transformer condition and operation parameters. Since the Transformer reacts to them, it can not filter or operate on them. There are two situations where you might want to do this. One is when you want to use incoming MIDI messages to change a Transformer's settings and the other is when you want meta-events to both set Transformer parameters and do something else. An instance when you might want to do this is illustrated in the patch "Safe Transpose" *figure 54.*



```
▽All Notes Off
(Fader)
Icon
  as Button
Out    Control
Channel       1
-1-         123
In      Meta
Channel       1
-1-         127
Range    0    0
Val as      Num
Filter      off
```

Fig. 54: Patch to turn all notes off before transposing

The Fader Transpose sends out meta-events which set the parameter of an add operation in the Transformer Pitch. The button-Fader All Notes Off receives the meta-event first and converts it to an all-notes-off controller message to avoid leaving hanging notes. (The Transformer Fix val=0 fixes the all-notes-off control message's value to 0 which is necessary to have the message work.)

4.10 The Audio Object – Fader & Instrument in One Package

If you are using the audio version of Logic, the last item on the environment's New menu is the Audio Object. If you select it you will create the Fader-like environment object similar to the one pictured below. Although this object does have some of the Fader's characteristics, it is really more like an Instrument. Its purpose is to manage audio input and output.

Since Logic supports a number of different hard disk recording systems, it has several species of Audio Object. The one pictured in *figure 55* relates to a Digidesign Audio Media II card. The other forms are pictured and described in your *Logic Audio manual.*

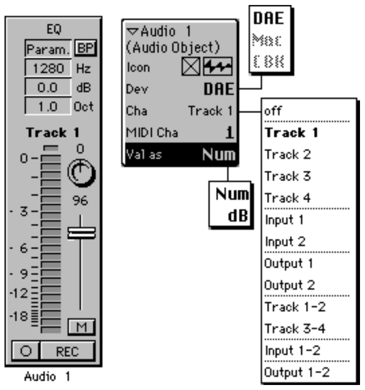


Fig. 55: The audio object, its parameters and menus

As you can see the Audio Object's parameter box lacks most of the familiar Fader parameters and has a couple of new parameters of its own. Below the familiar icon and Instrument menu checkbox, the Dev parameter allows you to select what audio source the object addresses. If you are using more than one source, you will create separate Audio Objects for each audio channel of each source.

The Cha parameter selects the audio track of your hard disk recording system addressed by the Audio Object. Changing the Cha setting automatically changes the MIDI Cha setting (see below). Usually you will want matching audio and MIDI channel numbers, but if not, beware to change the MIDI channel number back.

The MIDI Cha determines what MIDI channel the object will respond to. The Audio Object responds to the standard MIDI controllers for volume and pan (controllers #7 and #10 respectively). For EQ parameters, controller #16 controls frequency (labeled Hz), #17 controls bandwidth (labeled Oct), #18 controls gain (labeled dB) and #19 controls the bypass button (labeled BP). Controllers #20 thru #23 control EQ-2 on systems with two EQs available. Controllers #24 thru #31 control auxiliary sends when available. Consult your Logic Audio manual to learn the specifics of the Audio Objects in your system.

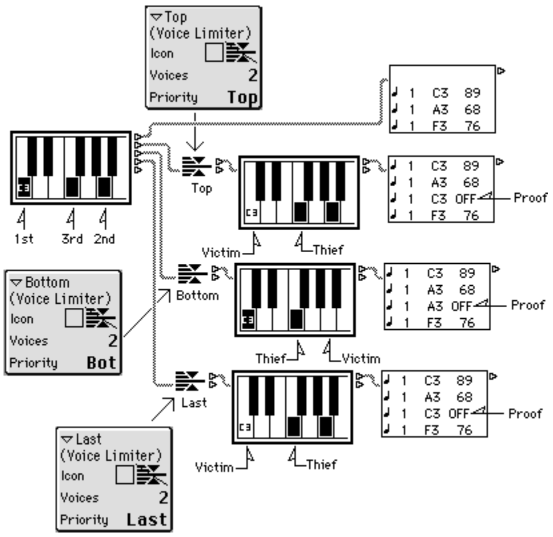
Although the Audio Object has no outlets, you can record the movement of Audio Object controls just like any other Fader. The only difference is that you must do so on a MIDI track, not an audio track. For playback, you will not want these controller events sent out over MIDI where they may well wreak havoc. Instead, you want them to be sent back to the Audio Object. The Instrument, A-Playback, in Logic's default song is cabled into all Audio Objects and *not* to any MIDI output port. Playback of MIDI controller events on a track using this as its instrument will affect the Audio Objects only.

The same thing applies to HyperDraw – do it on an A-Playback track.

4.11 Voice Limiting — High Crimes & Misdemeanors

Like the Transformer and Fader, the Voice Limiter acts on MIDI events passing through it. The Voice Limiter keeps a running total of the number of note-ons minus the number of note-offs passing through it – i.e. a running count of notes that are on – and generates note-offs to keep this total less than or equal to its Voices parameter setting.

The method of generating the note-offs is controlled by the Priority parameter setting (*figure 56*). In some circles, this is called note stealing. If the priority is Top then the lowest note currently on is turned off to make way for a new note. If the priority is Bot then the highest note is turned off and if the priority is Last then the earliest note (the one that's been on the longest) is turned off.



```
▽Top
(Voice Limiter)
Icon
Voices      2
Priority  Top

▽Bottom
(Voice Limiter)
Icon
Voices      2
Priority  Bot

▽Last
(Voice Limiter)
Icon
Voices      2
Priority  Last

1  C3  89
1  A3  68
1  F3  76

1  C3  89
1  A3  68
1  C3  OFF
1  F3  76

1  C3  89
1  A3  68
1  A3  OFF
1  F3  76

1  C3  89
1  A3  68
1  C3  OFF
1  F3  76
```

Fig. 56: Three kinds note stealing

The uses for this include:

•to simulate a solo instrument (voices=1)

•to limit the number of voices claimed by a channel on a multi-timbral MIDI sound device.

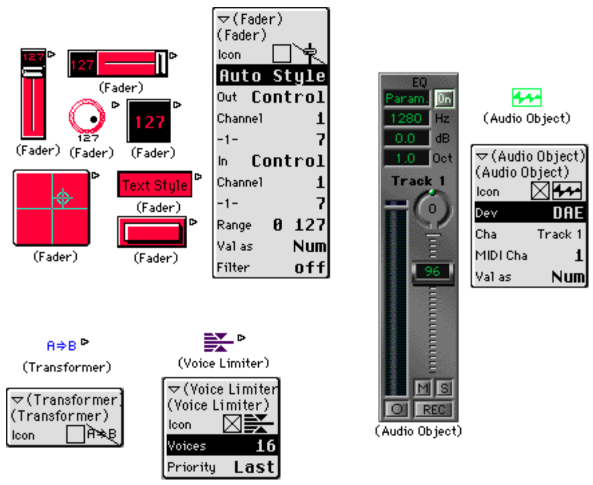
•to create standard MIDI files for systems with voice constraints (e.g. game-player sound cards)

The Voice Limiter is very easy to use. Simply cable it into any Logic Instrument and use it as a track instrument or cable it between any Logic Instrument and its output port. (Be sure to Remove the default port connection when doing this.)

There is one priority missing from the Voice Limiter. In some cases, you might want to prohibit any more note-ons from passing through until the note count falls below the limit. (This might be called first-note-priority.) The *Environment Toolkit* contains a patch for doing just this. The idea is to first create a "note-counter" (using meta-Faders and Transformers, what else?) then use it to control a Cable Switcher. This allows a separate route for each voice in the count and the simplest application is to leave dangling the outlets for voices above the desired voice limit.

Since the Voice Limiter automatically generates note-offs of the right pitch, it's tempting to try using it to prevent hung notes, for example when the pitch of incoming note-events is randomized. Unfortunately, the Voice Limiter is helpless when it comes to turning the last note off, so another method is needed. (In general you can not use the Transformers random operations on note pitches – Logic prohibits this to prevent hung notes – but there are several ways to subvert this. One is discussed in the section entitled *The Arpeggiator* below. Several others are presented in the *Environment Toolkit*.)

4.12 Running Status of the Human Kind – Part II



(Fader)
(Fader)
Icon
Auto Style
Out Control
Channel 1
-1- 7
In Control
Channel 1
-1- 7
Range 0 127
Val as Num
Filter off

(Audio Object)
(Audio Object)
Icon
Dev DAE
Cha Track 1
MIDI Cha 1
Val as Num

(Transformer)
(Transformer)
Icon

(Voice Limiter)
(Voice Limiter)
Icon
Voices 16
Priority Last

Fig. 57: Running status II - objects so far

We have introduced four new objects – the Transformer, Fader, Voice Limiter and Audio Object. The Voice Limiter and the Audio Object are fairly simple as each has one basic function. The Audio Object is a sort of an Instrument and Fader combined. Audio tracks must have an Audio Object as their instrument and that is their only function.

The Voice Limiter ensures that the number of held notes stays below a certain "threshold" level. It does this by voice stealing and it can be set to steal voices from the top, bottom or in chronological order. The one thing it can not do is inhibit new notes from sounding until the voice count drops below the threshold level.

The Transformer and Fader have some similarities and some differences. One similarity is that the they both change MIDI data flowing through them. One difference is that as screen objects, the Transformer is passive while the Fader is active – mousing the Fader creates data. Another difference is that the Fader can affect Logic by sending out meta-events and that Transformer parameters are one of the things meta-events can control.

When a MIDI event enters a Transformer four things can happen:

1) It can come out unchanged.
2) It can come out changed.
3) Both of the above.

4) None of the above (it gets thrown away).

MIDI events typically have four parts – status (type of event), MIDI channel (1-16) and two data values (between 0 and 127). But in some cases, there is only one data value. A Transformer can change any or all of these parts. (The one kind of MIDI data a Transformer can't change is SysEx data.) For the numerical parts – the channel and data values – the Transformer can perform many mathematical and logical operations like adding, multiplying, quantizing, flipping, limiting and user-defined mapping.

When a MIDI event enters a Fader, three things can happen:

1) It can come out unchanged.
2) It can come out changed.
3) None of the above (it gets thrown away).

A Fader can only change three parts of the incoming MIDI event – the status, channel and first data value – and it is limited to simply changing these parts to another fixed value (like the Transformer's Fix operation). The second data value changes the Fader rather than the other way around which means that the Fader "remembers" the last value it has seen. Since a Fader can set a Transformer's parameters using meta-events, a Transformer parameter can also be made to "remember" the last value seen. The difference is that a Transformer can use this "remembered" value in any number of ways while a Fader can only resend it (via the bang event) in its original form – the form matching the Fader's Out definition. When you simply want to resend the current Fader value, use a bang meta-event. Finally, there is the Fader's unique Thru filter mode which blocks incoming MIDI data but allows data coming from sequences or environment objects to pass.

Here are some rules-of-thumb for when to use a Transformer and when to use a Fader:

- When you just want to alter MIDI data, use a Transformer.
- When you want to create MIDI data with the mouse, use a Fader.
- When you want MIDI data displayed, use a Monitor, Keyboard or Fader according to the type of display desired.
- When you want to select or alter MIDI data based on other MIDI data arriving earlier or from a different source, use a meta-Fader/Transformer combination.

Chapter 5

It’s About Time

Each of the objects we’ve discussed so far is independent of Logic’s MIDI clock. Unless the input to a patch is coming from a sequence or the output is being recorded, the Transport needn’t be running for the patch to function. Put another way, once an event has started its passage through a patch, it proceeds as quickly as possible. Although the cabling in a patch controls the order in which things happen, the patch processing is virtually instant.

The remaining four objects, the MIDI Metronome Click, the Delay Line, the Arpeggiator and Touch Tracks™ all depend on Logic’s MIDI clock for the timing of their actions and all require that the Transport be running in order to function.

MIDI (as opposed to SMPTE) time displays in Logic typically contain four numbers: bar, beat, division and ticks (*figure 58*). This is the format in the Event editor and the Transport’s locator and position display. The time could have been indicated just in ticks but the four number display is much easier to read. The relation between bars and beats is the time signature. This can change during the song and the displays will change accordingly. The division is an arbitrary note size which determines the grid display in the Matrix editor and the increment amount in a various numerical displays that relate to time – for example, the Delay Line’s delay time parameter. The division is not tied to a song position and it can be changed for convenience at any time. The choice of division also determines where the tick count “rolls over”. The relation between ticks and note values throughout Logic is 960 ticks per quarter note.

There are 960 ticks per quarter note – this never changes.

The tempo determines the duration of a tick just as it does for a quarter note.

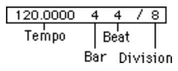


Fig. 58: How time is measured - bar/beat/div/tick

5.1 The MIDI Metronome Click

The purpose of the MIDI Metronome Click (a.k.a. the Metronome) is to generate MIDI note-events at the three clock divisions of Logic's time signature: beat, bar and division. The Metronome is like an Instrument in that it can have a built-in connection to a port and channel and/or it can be cabled to other environment objects.

There can be only one Metronome in the environment – if you create a new one from the menu when one already exists, the existing one will simply be moved to the current layer.

The Metronome is controlled by the Transport. If the Metronome icon is dark, the Metronome does not send out MIDI data. Logic remembers the Metronome's status separately for recording and playback. There are checkboxes for these states in the **Recording Options** portion of the **Song Settings** and these mimic and/or change the Transport button.

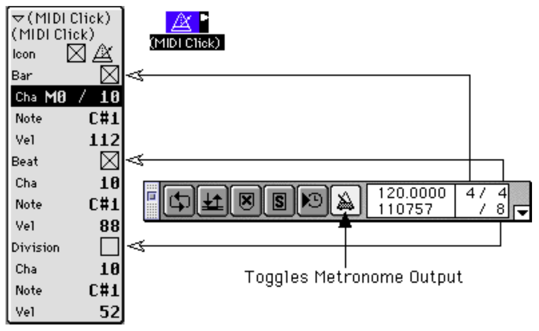

▽(MIDI Click)
(MIDI Click)
Icon ☒
Bar ☒
Cha M0 / 10
Note C#1
Vel 112
Beat ☒
Cha 10
Note C#1
Vel 88
Division ☐
Cha 10
Note C#1
Vel 52

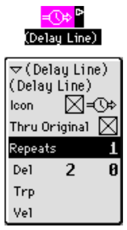
Fig. 59: The metronome object and its parameters & transport controls

The Metronome object has separate channel, note and velocity settings for each of its divisions so you can be fairly creative about how the Metronome sounds. Also there are **Recording Options** preferences for **Speaker Clicks** which causes the Metronome to sound through the computer's internal speaker and for **Polyphonic Clicks**. If Polyphonic Clicks is enabled, then the smaller division clicks will sound at the larger divisions – otherwise, the beat notes won't be sent at the bars and the division notes won't be sent at the beats or bars.

You can use the Metronome output to trigger other environment processes but this option is somewhat limited if you also want to use the Metronome for its intended purpose. Notice, however, that the output of the Metronome is functionally equivalent to a looped, one-measure sequence with notes on the bar, beat and division positions.

5.2 The Delay Line

The Delay Line (*figure 60*) repeats incoming MIDI events at regular intervals. The delay interval is measured in divisions & ticks and the Transport must be running for the delay to work.


(Delay Line)

▽ (Delay Line)
(Delay Line)
Icon ☒
Thru Original ☒
Repeats 1
Del 2 0
Trp
Vel

Fig. 60: Delay line object and parameter box

The Thru Original parameter determines whether the original event is passed through to the output. You might not want this to be the case if the delays are echoed to a different sound than the original.

The Repeats parameter determines how many MIDI events follow the original. Its range is 0 to 99 and each new event is delayed from the preceding one.

The Del parameter sets the delay time. The leftmost number is divisions and the rightmost number is ticks. The ticks parameter "rolls over" by incrementing the divisions parameter. The delay time stays fixed when you change Logic's format and the Del display will therefore change. (You must deselect and re-select the Delay Line object or click its Del parameter to see this change.)

The Trp and Vel parameters are added cumulatively to each new event created by the delay object. Their ranges are -99 to 99. A Trp value of 1, for example, will cause the delayed notes to climb by semitones.

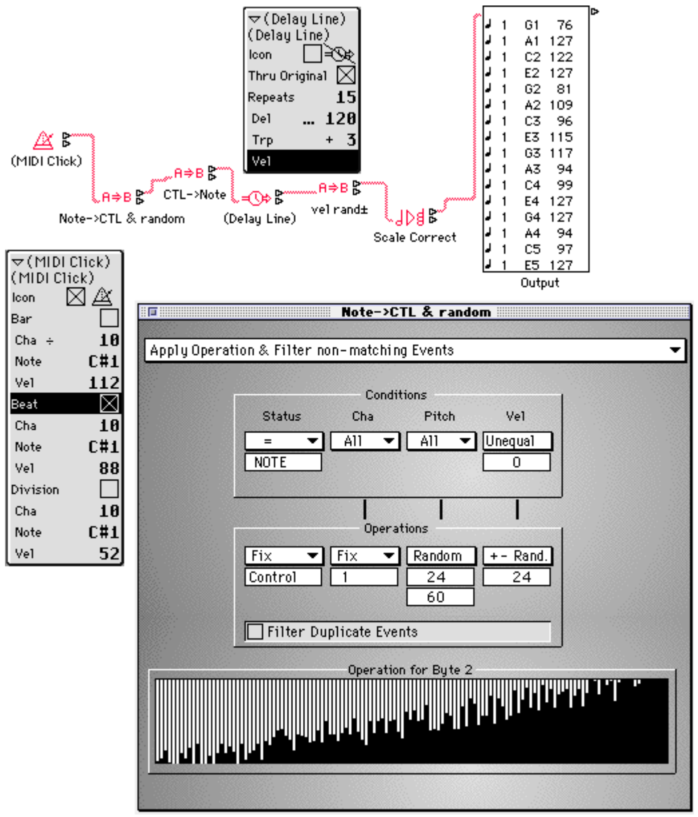
ℹ The Delay Line acts on all types of MIDI events, not just notes, so only tell it what you want repeated.

The repeated notes have the same length as the incoming note. As a result, sustained notes with a fast delay and many repeats use up MIDI sound device voices very fast – keep this in mind when creating patches with the Delay Line.

There are many creative ways to use the Delay Line. In both of these examples we use the Metronome to "feed" the Delay Line. This is a convenient way to build and debug Delay Line patches but ultimately, a sequence is the best source of input. If a metronomic result is intended, use a 1 bar sequence with a single quarter note at the beginning and loop it.

Delay Line Arpeggiator

The patch shown in *figure 61* incoming note-ons (generated by the Metronome in this case) are converted to control events so that the pitch and velocity can be randomized. They are then converted back to notes and passed to the Delay Line. The Delay Line sends out the original and 15 repeats at 240 tick (16th note) intervals. Each repeat is transposed up a minor 3rd producing a diminished scale. From the Delay Line, the notes pass through another Transformer where their velocity is again randomized. This is to add a velocity contour to the arpeggio since all notes leaving the Delay Line have the same velocity. Finally, the notes pass through the Chord Memorizer which maps all black keys up to the neighboring white key (i.e. it corrects to the key of C). The result is an arpeggio of major and minor thirds in the key of C starting on a random pitch.



(Delay Line)
(Delay Line)
Icon
Thru Original ☒
Repeats 15
Del … 120
Trp + 3
Vel

```
1  G1   76
1  A1  127
1  C2  122
1  E2  127
1  G2   81
1  A2  109
1  C3   96
1  E3  115
1  G3  117
1  A3   94
1  C4   99
1  E4  127
1  G4  127
1  A4   94
1  C5   97
1  E5  127
```

(MIDI Click)
(MIDI Click)
Icon ☒
Bar ☐
Cha ÷ 10
Note C#1
Vel 112
Beat ☒
Cha 10
Note C#1
Vel 88
Division ☐
Cha 10
Note C#1
Vel 52

Note->CTL & random
Apply Operation & Filter non-matching Events

Conditions

Status	Cha	Pitch	Vel
=	All	All	Unequal
NOTE			0

Operations

Fix	Fix	Random	+- Rand.
Control	1	24	24
		60	

☐ Filter Duplicate Events

Operation for Byte 2

Fig. 61: Delay line arpeggiator patch

Try replacing the key correcting Chord Memorizer with a key filtering one (i.e. black keys mapped to nothing).

Do not feed this patch with live MIDI input – hung notes will result. (The *Environment Toolkit* contains a solution for this.)

Multi-Tap

If multiple outlets of the Delay Line are cabled, the repeats will be spread among them. If two outlets are used then the repeats will alternate between them. If the same number of outlets are used as there are repeats (including the original if Thru Original is checked) then each repeat will have its own outlet. Individual repeat parameters can then be controlled separately.

In *figure 62*, each repeat is cabled through a Transformer which controls its pitch. There is a meta-Fader cabled to each Transformer to set the pitch parameter. Finally, all Transformers are cabled through an output Transformer which blocks notes of pitch C-2 (note number 0). Thus, when a meta-Fader / Transformer combination sets a repeat's pitch to C-2, that repeat is silenced.

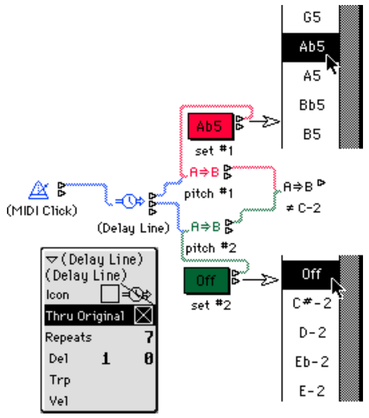


Fig. 62: Delay line multi-tap - two tap schema

Figure 63 expands the example to 8 taps. Although the Metronome is shown feeding the patch, live input produces more interesting results. Notice the resemblance of this patch to an analog sequencer.

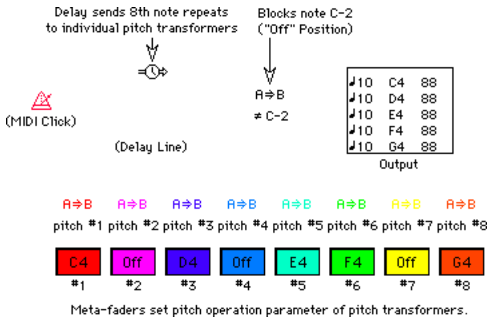
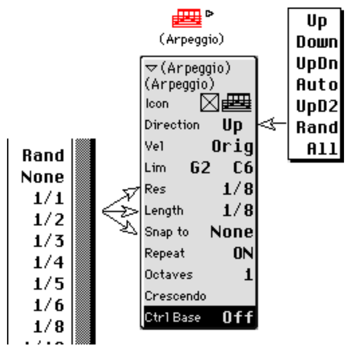


Fig. 63: Delay line multi-tap expanded to 8 steps

The Delay Line can be used to create any kind of MIDI events. In the *Environment Toolkit* we use Delay Lines to create several styles of MIDI LFOs and analog-style sequencers.

5.3 The Arpeggiator

Like the Delay Line, the Arpeggiator's action depends on Logic's MIDI clock and the Transport must be running for it to have any effect. The Arpeggiator (*figure 64*) acts on the currently held notes, repeating them in a pattern determined by its parameter settings.



```
(Arpeggio)
(Arpeggio)
Icon
Direction   Up
Vel       Orig
Lim     G2  C6
Res        1/8
Length     1/8
Snap to   None
Repeat      ON
Octaves      1
Crescendo
Ctrl Base  Off
```

Fig. 64: Arpeggiator object with parameters & menus

The Repeat parameter determines whether the arpeggio is played once (blank) or repeated as long as the notes are held (ON).

The Direction parameter sets the direction of the arpeggio. In the Auto setting the result depends on whether the lowest or highest of the held notes arrived first – it is the same as Up if the lowest note arrived first and the same as Down otherwise. UpDn arpeggiates the notes from bottom to top then from top to bottom, repeating the top and bottom notes at the turns. UpD2 is the same as UpDn except that the bottom and top notes are not played twice at each turn. Rand plays the notes in random order. (Sample & hold, anyone?) All repeats them as a chord.

The Octaves parameter sets the number of octaves over which the arpeggio is extended. Think of the arpeggio as a sequence constructed from a "bag" of notes. If the octave is set to 1, the bag contains only the currently held notes. If the octave is set to 2, the bag contains two notes for each held note – itself and a note an octave higher. If the octave is set to 3, the bag contains three notes for each held note, etc. The direction parameter applies to all notes in the bag. For example, if F3 and G4 are held with the octave set to 2 and the direction set to Up then the arpeggio will be: F3 F4 G4 G5 (not F3 G4 F4 G5).

The velocity and crescendo parameters affect the velocities of the arpeggiated notes. The velocity settings are Orig for the original note velocity, Rand for a randomly assigned velocity or a number from 1 to 127 for a fixed velocity. The crescendo parameter only applies when repeat is on. In this case the crescendo amount is added to each note's velocity cumulatively at each repeat. The crescendo range is -99 to 99.

The Lim parameter limits the incoming note range. Notes outside the range are passed through but not used in the arpeggio.

The Res, Length and Snap to parameters together determine the timing of the arpeggio. Res is the time between arpeggio notes. Length is the note duration. Snap to delays the start of the arpeggio to the next SPL position corresponding to its setting. (A setting of None causes the arpeggio to start immediately.) The Snap to parameter has no Rand choice. For Length, None is replaced by Orig for the original (incoming) note length. This is only meaningful when a sequence feeds the Arpeggiator – for MIDI input, the note length is the time to the next occurrence of the same note. A Res setting of None means instant – the only effect is octave doubling if the octaves parameter is greater than 1.

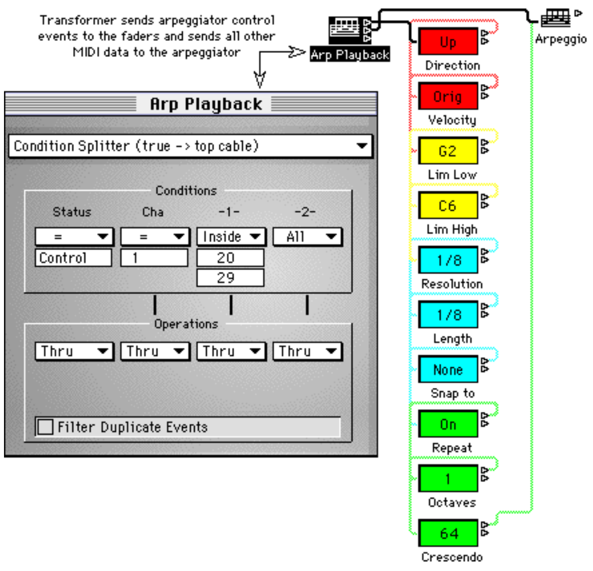
When the transport is running, you can use a Transformer's "Random" and "+-Rand" operations to randomize the pitches of notes coming from an Arpeggiator. If you turn "Repeat" off and set "Octaves" to 1, this will allow you to randomize pitches but all notes will have the same length.

The Ctrl Base parameter allows MIDI controller events to change the other Arpeggiator parameters (similar to meta-events for Transformers). Settings other than Off determine the controller number that affects the direction parameter. Successive controller numbers control the remaining parameters down the list. For example, if Ctrl Base is set to 20 then controller #21 affects the Vel parameter, controller #22 affects the left Lim parameter, etc.

Note that the Arpeggiator is a one-channel object – if you feed it a multi-channel chord, it will output every thing on the channel of the last note received.

5.4 A MIDI Controlled Arpeggiator

The patch in *figure 65* is handy both for experimenting with Arpeggiator settings and for recording and storing Arpeggiator parameter presets. The ten text Faders control the ten MIDI controllable Arpeggiator parameters. The Transformer named Arp Playback splits incoming MIDI data between the controller events that affect the Arpeggiator and everything else. The controller events are sent to the Faders and the rest is sent directly to the Arpeggiator. Note events feed the Arpeggiator and other MIDI events pass through unchanged.


Transformer sends arpeggiator control events to the faders and sends all other MIDI data to the arpeggiator

Arp Playback

Condition Splitter (true -> top cable)

Conditions

Status	Cha	-1-	-2-
=	=	Inside	All
Control	1	20	
		29	

Operations

Thru	Thru	Thru	Thru

Filter Duplicate Events

Up	Direction
Orig	Velocity
G2	Lim Low
C6	Lim High
1/8	Resolution
1/8	Length
None	Snap to
On	Repeat
1	Octaves
64	Crescendo

Fig. 65: MIDI controller arpeggiator - schematic view

To set up the ten Faders:

- Create a new text style Fader and set its In and Out definitions to match the Arpeggiator’s Ctrl Base setting (controller #20 in this example).

- Make nine copies of the text Fader.

- Select the original text Fader and from the **Options** menu, choose **Apply Buffer Template to**. Then from the sub-menu, choose **Define Template**.

- Select all ten text Faders and from the **Options** menu, choose **Apply Buffer Template to**. Then from the sub-menu, choose **Definition, number increment**. This sets the ten Faders’ In and Out definitions to controllers #20 thru #29.

- With all ten Faders still selected, choose **Cable serially** from the **Options** menu.

- Cable the Transformer Arp Playback and the Arpeggiator (named Arpeggio) as in the illustration.
- Name the Faders as in the illustration and enter text to match the parameters they control.

To use the patch, choose Arp Playback as a track instrument. When the Transport is running, MIDI input or sequences played on this track will be arpeggiated. Channel 1 controller #20 thru #29 events will change the Arpeggiator's parameters. To record Arpeggiator parameter settings, select the appropriate Faders, enable recording in pause mode and choose **Send Selected Fader Values** from the **Options** menu.

To create Arpeggiator presets, record the Fader values as above then copy & paste them into SysEx buttons. Cable the output of the SysEx button to the Arp Playback Transformer.

Figure 66 shows a more eye-pleasing version of the MIDI controlled Arpeggiator. Some presets have been added and a Transformer has been placed after the Arpeggiator to filter out notes with velocity 1. The Echo preset shown repeats all held notes as 8th note triplets (1/12th notes) decreasing the velocity by 8 on each repeat. When the velocity reaches 1, the note is blocked and the echo stops.

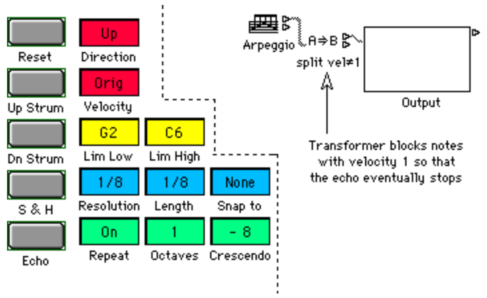
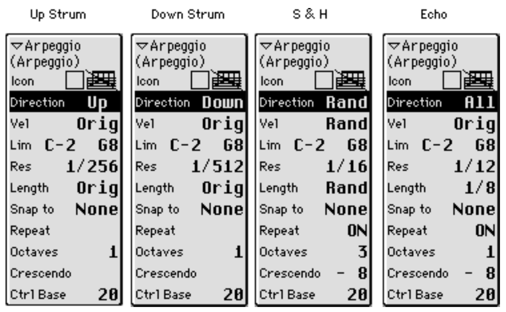


Fig. 66: Eye pleasing view

The Reset preset returns the Arpeggiator to its default settings. The Up Strum and Dn Strum presets (*figure 67*) are effective when alternated using a guitar-like sound. The S & H (sample & hold) preset randomly selects notes from the currently held notes. Because of the octaves parameter setting the sampled notes may be raised one or two octaves. The crescendo setting of -8 cause the output to eventually stop as in the Echo preset.



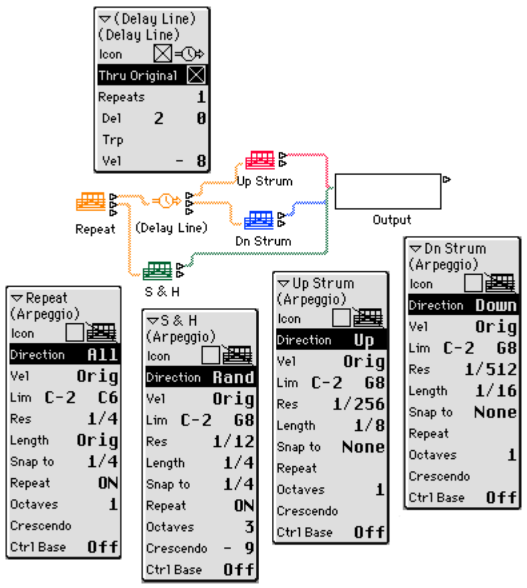
	Up Strum	Down Strum	S & H	Echo
	▽Arpeggio (Arpeggio)	▽Arpeggio (Arpeggio)	▽Arpeggio (Arpeggio)	▽Arpeggio (Arpeggio)
Icon				
Direction	Up	Down	Rand	All
Vel	Orig	Orig	Rand	Orig
Lim	C-2 G8	C-2 G8	C-2 G8	C-2 G8
Res	1/256	1/512	1/16	1/12
Length	Orig	Orig	Rand	1/8
Snap to	None	None	None	None
Repeat			ON	ON
Octaves	1	1	3	1
Crescendo			- 8	- 8
Ctrl Base	20	20	20	20

Fig. 67: Presets' parameter settings

5.5 A Strummer with Random Accents

Arpeggiators and Delay Lines can be combined to produce many interesting effects. One example is the strummer shown in *figure 68*. It can be used by either cabling an Instrument into the Arpeggiator named Repeat or by using Repeat for a track's instrument. The Monitor, Output, should be cabled to the a port or into some other environment patch leading to a port.

Chords appearing at the Arpeggiator, Repeat, are "snapped" to the next quarter-note and repeated at quarter-note intervals. They are sent to both the Delay Line and the Arpeggiator, S & H. The Delay Line sends the original to the Arpeggiator Up Strum which creates a fast, one-time arpeggio from low note to high note. An 8th-note (480 ticks) later, the first (and only) repeat from the Delay Line is sent to the Arpeggiator Dn Strum which creates a faster, one-time arpeggio from high note to low note. The first strum is an 8th-note in length and the second is a 16th-note and its velocity is reduced by 8. The effect is alternating up and down 1/8th-note strumming until the chord is released.



```
▽(Delay Line)
(Delay Line)
Icon
Thru Original  ☒
Repeats        1
Del      2     0
Trp
Vel          - 8
```

```
▽Repeat
(Arpeggio)
Icon
Direction   All
Vel        Orig
Lim   C-2    C6
Res         1/4
Length     Orig
Snap to     1/4
Repeat       ON
Octaves       1
Crescendo
Ctrl Base   Off
```

```
▽S & H
(Arpeggio)
Icon
Direction  Rand
Vel        Orig
Lim   C-2    G8
Res        1/12
Length      1/4
Snap to     1/4
Repeat       ON
Octaves       3
Crescendo   - 9
Ctrl Base   Off
```

```
▽Up Strum
(Arpeggio)
Icon
Direction    Up
Vel        Orig
Lim   C-2    G8
Res       1/256
Length      1/8
Snap to    None
Repeat
Octaves       1
Crescendo
Ctrl Base   Off
```

```
▽Dn Strum
(Arpeggio)
Icon
Direction  Down
Vel        Orig
Lim   C-2    G8
Res       1/512
Length     1/16
Snap to    None
Repeat
Octaves       1
Crescendo
Ctrl Base   Off
```

Fig. 68: Arpeggiator as strummer with random accents

The Arpeggiator Repeat also sends the chord to the Arpeggiator S & H, which sends out randomly selected notes from the chord within a 3 octave range and with velocity reduced by 9. These are snapped to the quarter-note and sent out as 8th-note triplets (1/12 resolution) which results in a sample & hold like melody in 12/8 over the strum in 4/4. The patch could be further automated by using a Chord Memorizer before the Arpeggiator Repeat.

5.6 Touch Tracks — The Final Frontier

Logic's Touch Tracks™ object is a triggering device. It allows you to use MIDI note-events to initiate playback of individual sequences or folders. Anything that can be put on a track in Logic's Arrange window can be put on a Touch Tracks EXCEPT audio – Touch Tracks only works for MIDI data. Since you can select everything in the Arrange window and pack it into a folder, you can assign the MIDI portion of any Logic song to a Touch Tracks key. Since a sequence can consist of a single MIDI event, Touch Tracks can trigger single events (though there's not much point in doing so).

Touch Tracks objects (*figure 69*) bear a visual resemblance to Mapped Instruments but the resemblance is only superficial. The Touch Tracks' parameter box, for example, is a vestige left over from the Mapped Instrument and has no meaning for a Touch Tracks – you can ignore it. The Touch Tracks object has a cable outlet but this also has no function and can be ignored. Both of these will disappear in future versions.

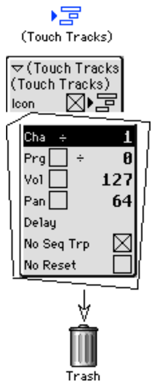
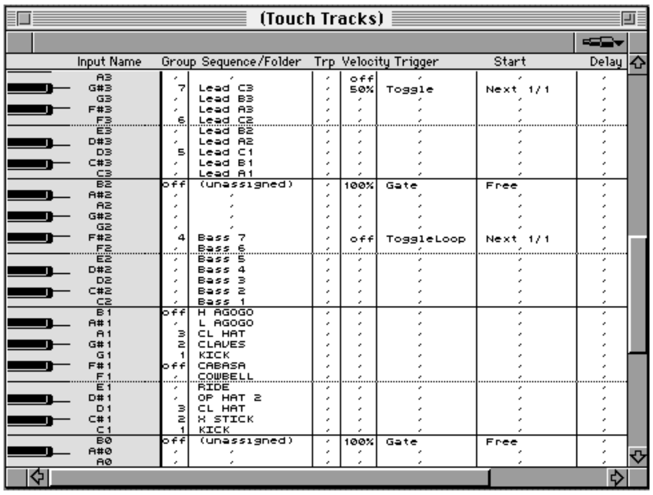


Fig. 69: Touch tracks & parameter box

When you create a Touch Track, a window opens up (*figure 70*) which is similar to the Mapped Instrument's window and this is where all the action is. You assign sequences or folders to Touch Tracks keys by dragging them to the appropriate line in the Touch Tracks window.

The Touch Tracks window has a keyboard on the left and eight columns of playback parameters. When you drag a sequence to a row in the window, the key for that row triggers the sequence. In order for this to happen, the Transport must be running and the Touch Tracks must be the instrument for the currently selected track in the Arrange window.



(Touch Tracks)

Input Name	Group	Sequence/Folder	Trp	Velocity	Trigger	Start	Delay
A3	ʼ	ʼ	ʼ	off	ʼ	ʼ	ʼ
G#3	7	Lead C3	ʼ	50%	Toggle	Next 1/1	ʼ
G3	ʼ	Lead B3	ʼ	ʼ	ʼ	ʼ	ʼ
F#3	ʼ	Lead A3	ʼ	ʼ	ʼ	ʼ	ʼ
F3	6	Lead C2	ʼ	ʼ	ʼ	ʼ	ʼ
E3	ʼ	Lead B2	ʼ	ʼ	ʼ	ʼ	ʼ
D#3	ʼ	Lead A2	ʼ	ʼ	ʼ	ʼ	ʼ
D3	5	Lead C1	ʼ	ʼ	ʼ	ʼ	ʼ
C#3	ʼ	Lead B1	ʼ	ʼ	ʼ	ʼ	ʼ
C3	ʼ	Lead A1	ʼ	ʼ	ʼ	ʼ	ʼ
B2	off	(unassigned)	ʼ	100%	Gate	Free	ʼ
A#2	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ
A2	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ
G#2	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ
G2	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ
F#2	4	Bass 7	ʼ	off	ToggleLoop	Next 1/1	ʼ
F2	ʼ	Bass 6	ʼ	ʼ	ʼ	ʼ	ʼ
E2	ʼ	Bass 5	ʼ	ʼ	ʼ	ʼ	ʼ
D#2	ʼ	Bass 4	ʼ	ʼ	ʼ	ʼ	ʼ
D2	ʼ	Bass 3	ʼ	ʼ	ʼ	ʼ	ʼ
C#2	ʼ	Bass 2	ʼ	ʼ	ʼ	ʼ	ʼ
C2	ʼ	Bass 1	ʼ	ʼ	ʼ	ʼ	ʼ
B1	off	H AGOGO	ʼ	ʼ	ʼ	ʼ	ʼ
A#1	ʼ	L AGOGO	ʼ	ʼ	ʼ	ʼ	ʼ
A1	3	CL HAT	ʼ	ʼ	ʼ	ʼ	ʼ
G#1	2	CLAVES	ʼ	ʼ	ʼ	ʼ	ʼ
G1	1	KICK	ʼ	ʼ	ʼ	ʼ	ʼ
F#1	off	CABASA	ʼ	ʼ	ʼ	ʼ	ʼ
F1	ʼ	COWBELL	ʼ	ʼ	ʼ	ʼ	ʼ
E1	ʼ	RIDE	ʼ	ʼ	ʼ	ʼ	ʼ
D#1	ʼ	OP HAT 2	ʼ	ʼ	ʼ	ʼ	ʼ
D1	3	CL HAT	ʼ	ʼ	ʼ	ʼ	ʼ
C#1	2	X STICK	ʼ	ʼ	ʼ	ʼ	ʼ
C1	1	KICK	ʼ	ʼ	ʼ	ʼ	ʼ
B0	off	(unassigned)	ʼ	100%	Gate	Free	ʼ
A#0	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ
A0	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ	ʼ

Fig. 70: Touch tracks window

a Touch Tracks uses all the sequence and track parameters of the source sequence or folder. These determine what instrument plays the sequence, what environment processing it suffers, etc.

In addition to real time triggering of sequences and folders, a Touch Tracks can be used as an arranging tool. After the constituent sequences have been assigned to Touch Tracks keys, create a sequence on the Touch Tracks track in the Arrange window and manipulate the data in this sequence to alter the order and timing of the sequences in the Touch Tracks.

Since the sequences and folders triggered by a Touch Track also exist in the Arrange window, you will usually want to mute them. This has no effect on Touch Tracks playback – Touch Tracks ignores the mutes.

It's a good idea to Collect all sequences and folders assigned to touch tracks into a single folder and mute that folder.

Another use for Touch Tracks is to create MIDI controller LFO effects. Start by creating a sequence containing one "cycle" of controller data – a volume ramp for example – then assign the sequence to a Touch Tracks key in GateLoop or ToggleLoop mode. In GateLoop mode, the LFO sequence loops as long as the note which triggers it is held down. In ToggleLoop mode, the first occurrence of the note starts the LFO sequence looping and the next occurrence stops it.

Here's a detailed look at how to use a Touch Track:

Assigning Sequences To One or Several Notes

In the left column of the Touch Tracks window, you select a target key by clicking on it in the keyboard illustration. You can add keys to the selection by shift-clicking them and you can rubber-band select continuous groups of keys.

Once one or more keys are selected, dragging any sequence or folder from the Arrange window to the row of any one of them assigns that sequence or folder to all the selected keys. If more than one key is selected, the transpose parameter (see below) for each key above the target key is incremented and the transpose parameter for each key below the target key is decremented. The transpose parameter for the target key is left blank (meaning no transpose). For example, if you selected all the keys and dragged a sequence to the C3 row then C#3 would have a transpose of +1, D3 or +2, C4 of +12 etc. And, B2 would have a transpose of -1, Bb2 of -2, C2 of -12, etc. In fact, if you simply drag a sequence from the Arrange window to a blank space in the Environment window, a Touch Track will be created with just this configuration.

Notice that the keys don't have to be adjacent. If you select C2, C3 and C4 then drag a sequence to the C3 row, it will also be assigned to C2 with a transpose of -1 and to C4 with a transpose of +1.

How the Columns Work

In a new Touch Track, the top row (labeled G8) contains default settings in each column and each lower row contains an apostrophe in each column. The apostrophe means that the setting is the same as for the row above.

In operation, this takes a little getting used to because each time you change a column entry, the entry for the column below it will most likely change to show its actual value. For example the Group column default entry in row G8 is Off. If you change the group setting for row E8 then the display for row D#8 will change from the apostrophe to Off to indicate that this row's setting did not change.

The Group Column

Sequences or folders assigned to Touch Tracks rows can be grouped together so that their playback is mutually exclusive – i.e. only one row in the group can be triggered at a time. Triggering a row in the same group as one which is currently playing, causes the one that's playing to stop.

There are 99 possible groups as well as Off. Rows in different groups and rows whose group is Off can be triggered simultaneously.

The Sequence/Folder Column

This column shows the name of the sequence or folder assigned to the row. If nothing is assigned, it displays "(unassigned)" or an apostrophe. The names here are actively connected to the sequences and folders – they will change in real-time if the sequence or folder name is changed.

The Transpose Column

The Trp column determines how the sequence or folder (i.e. all sequences in the folder) is transposed when triggered by the Touch Tracks key. The default transpose is blank (no transpose) except, as mentioned above, if a sequence is dragged to several selected notes at once.

The Velocity Column

The setting here determines how much the playback velocities are affected by the velocity of the triggering note. The three choices are 100%, 50% and Off. 100% means "a lot", 50% means "not a lot" and Off means "not at all".

The Trigger Column

Three things can happen when you trigger a sequence from a Touch Tracks: It can play to the end (Multi and Single modes), it can play until the trigger-key is let up (Gate and GateLoop modes) or it can play until the trigger-key is let up and pressed again (Toggle and ToggleLoop modes). The difference between multi and single mode is that in multi mode, a new trigger will cause the sequence to start again without turning off currently playing versions whereas in single mode, each new trigger starts the sequence over (halting all currently playing versions). In the loop versions of the gate and toggle modes, sequence or folder playback loops until the event (gate-off or re-trigger) which stops it.

The Start Column

The triggering of a Touch Tracks event can be quantized to the next 16th, quarter or whole note by the setting in this column. The default setting of Free means that playback starts immediately. The start setting also applies to stopping the sequence – if for example, you have the trigger set to gate mode and the start set to whole notes then releasing the trigger note will stop the sequence at the next whole note.

The Delay Column

The triggering of a Touch Tracks event can be delayed by the setting in this column. Clicking the left part of the field pops a menu with the familiar, note-fraction choices. Clicking on the right side allows you to set the delay in ticks. This is how to ensure, for example, that playback starts on the third 8th-note of a measure (i.e. Start=1/1 and Delay=1440 ticks – three 8th-notes).

5.7 Running Status of the Human Kind – Part III

We've completed our survey of the environment by introducing four real-time objects: the MIDI Metronome Click, the Delay Line, the Arpeggiator and Touch Tracks™. They are called "real-time" objects because Logic's clock (i.e. the Transport) must be running in order for them to operate. Except for the Metronome, cycle jumps also interrupt their operation.

The MIDI Metronome Click generates note events at three different intervals. The first two intervals, the bar and beat, are determined by the active time signature – when the time signature changes in a Logic song, the Metronome's bar and beat clicks change accordingly. The third interval, the division, is determined by the current time grid which is indicated by the number after the "/" in all of Logic's time signature displays. The grid is not a function of song position – it is a user-set parameter that applies to the whole song but can be changed at any time.

The Delay Line reacts to incoming MIDI events by repeating them at a fixed interval. The number of repeats and the repeat interval are set in the Delay Line's parameter box. There are also parameters for passing or blocking the original event, for transposing notes and for adding or subtracting from their velocity. Finally, if multiple outlets are cabled from the Delay Line, the out coming events will be cycled through the cables from top to bottom and if there are more repeats than cables, the cycle starts again from the top.

The Arpeggiator reacts to incoming groups of note-events (chords) by playing them one-at-a-time. The order of playback of the currently held notes as well as the rate, note-length and repeat mode are set in the Arpeggiators parameter box. All the Arpeggiator's parameters can be controlled by MIDI controller events either from environment Faders or incoming MIDI data.

The Touch Tracks™ is a triggering device for sequences already recorded in Logic songs. Anything that can be put on a track in the Arrange window can be assigned to a Touch Track key except audio data. To use a Touch Track, assign it as a track's instrument in the Arrange window. Then use note events in sequences on this track or select the track and use MIDI input to trigger Touch Tracks events.

Chapter 6
Techniques

6.1 Recording Environment Processes

You will often want to incorporate the output of an environment process into a song. There are several ways to do this and the best choice usually depends on the context. One approach is to place the environment patch between Physical Input and to Sequencer (*figure 71*). This only works for processing MIDI input – velocity scaling for a MIDI keyboard, for example.

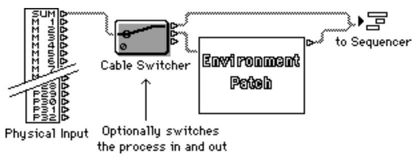


Fig. 71: Between PI and TS objects

A second method is to record the input data in a sequence then use the patch's input as the track's instrument (*figure 72*). This method is typically the best way to do automated mixing, for example, since Logic automatically records all Fader movements when in record mode.

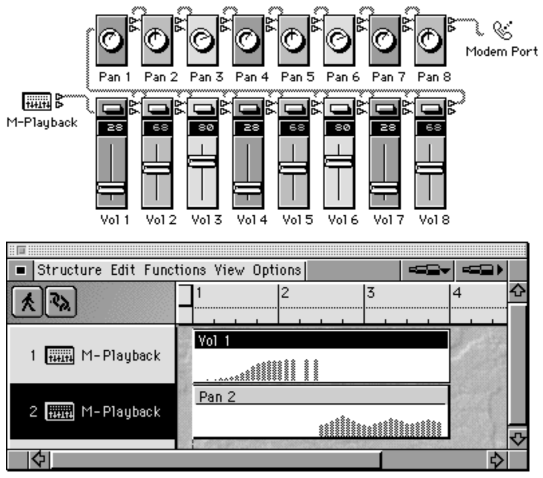


Fig. 72: As track instrument

Recording the data to be processed has two drawbacks. First, it doesn't allow the same environment patch to be reused since it is taken up processing the recorded data each time the song is played. Second, if there are any random aspects to the patch, you can not guarantee a specific outcome – i.e. you can't choose the best "take".

A third method is to first record the data which the patch processes then record the output of the patch (*figure 73*). With this method it is convenient to have a Cable Switcher to toggle the patch's output between the desired playback Instrument (or port or patch) and to Sequencer. The reason is that, when recording the input sequence, you will want to hear the environment process (switch in position 1). On the other hand, when recording the patch's output on a track using the playback Instrument, you will not want the environment patch also connected to this Instrument as this would result in double output (switch position 0). After you've finished recording the patch's output, use switch position 2 to silence the patch (or mute the input sequence).

As an alternative to the cable switch, you could record on a No Output track then move the recorded output to a Playback Instrument track but this is not recommended since the patch's output would always be cabled to to Sequencer and appear at the currently selected track.

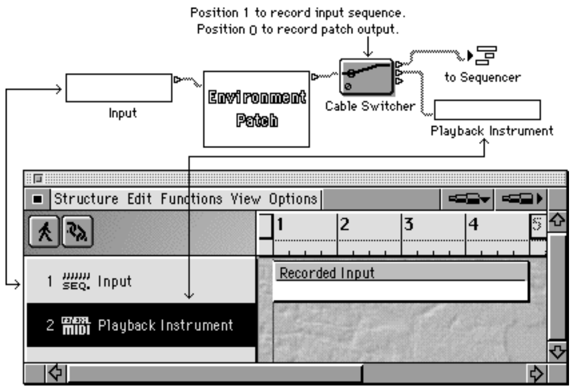

Position 1 to record input sequence.
Position 0 to record patch output.
to Sequencer
Input
Environment Patch
Cable Switcher
Playback Instrument
Structure Edit Functions View Options
1 2 3 4 5
1 SEQ. Input
2 MIDI Playback Instrument
Recorded Input

Fig. 73: Recording patch output

6.2 The Environment Menus

The Environment window has five menus: **New**, **Edit**, **View**, **Options** and **Import**. The Import menu is toggled on and off by selecting **Import Options** from the **View** menu. Many of the menu selections can be assigned to key commands. In addition to the menu selections, there are assignable environment key-commands for nudging the position and size of the selected object(s). There are key-commands for most commonly repeated environment tasks and key-commands are being added with each new release.

The New Menu

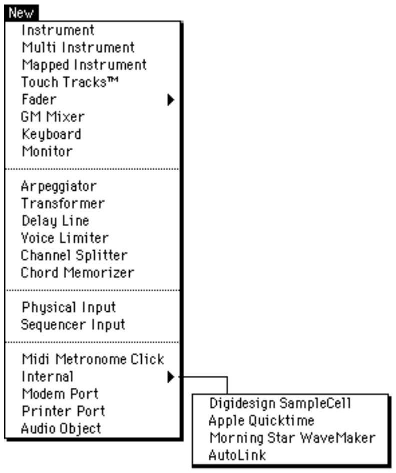


Fig. 74: The New menu & its sub-menus

The **New** menu is used for creating environment objects. It has one sub-menu for selecting Fader styles and types (see the heading *Faders, Faders, Faders* for a discussion of this sub-menu) and another for selecting **Internal** objects.

An internal object is a MIDI sound device installed inside your computer which is accessed internally rather than via MIDI. Three such internal devices are supported – Digidesign's SampleCell, Apple's Quicktime Musical Instruments and Morning Star's WaveMaker. Since these devices are not accessed via MIDI, a special, internal port is needed for Logic to communicate with them. The **Internal** sub-menu is used to create such objects.

Accessing internal devices is often much faster than the MIDI serial rate and in any case, it does not take up MIDI bandwidth.

The Edit menu

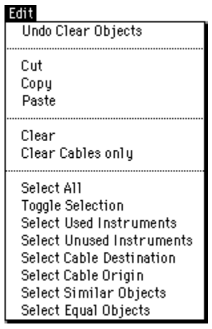


Fig. 75: The Edit menu & its sub-menus

Aside from the usual cut, paste and undo options, the **Edit** menu provides for selecting environment objects in various useful ways.

For the **Select Used Instruments** and **Select Unused Instruments** options, "used" includes any object which is a non-empty track's instrument *and* all objects that can be reached from it by following cables.

For the **Select Similar** and **Select Equal** options there is only a difference for Faders. In this case all Faders are similar but only Faders of the same style are equal.

Select Cable Destination is very handy for tracing environment patches. If you use this command when an object is selected, the first (top) cable will be followed. If you select one of an objects cables, that cable will be followed.

Clear Cables Only is useful when creating new objects by ⌥-dragging or copy & pasting. When you duplicate an object this way, all of the cables leading out of it are also duplicated and you usually don't want them. (The cables leading into the object are not duplicated.)

The View Menu

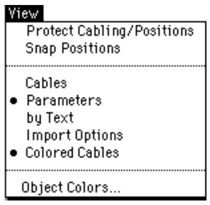


Fig. 76: The View menu & its sub-menus

The **View** menu controls the appearance of the Environment window. A choice is on when it is preceded by a "•".

The **Protect Cabling/Positions** option prevents any change in an object's position or cabling. The **Cables** option toggles display of cables and outlets. When **Cables** is off (hidden cables) and **Protect Cabling/ Positions** is on (no changes) the environment changes to a more pleasing shaded background.

The **by Text** option changes from a graphic display of objects to a list of objects by name. (The list is in the order of the objects' unique IDs).

The **Import Options** option toggles the Import menu on and off. See the sub-heading *Importing Environments from Other Logic Songs* below.

The **Colored Cables** options toggles cables between gray and colored – the cable color is taken from the object the cable comes from.

The **Object Colors** option allows you to select the color for all currently selected objects.

The Options Menu

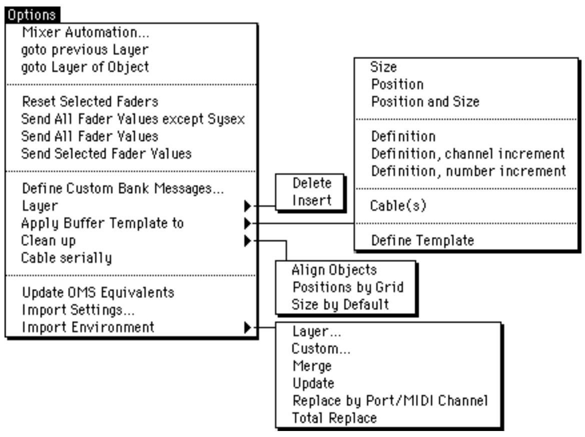


Fig. 77: The Options menu & its sub-menus

The environment's **Options** menu is its most extensive and most useful menu. It presents many options for both constructing and using environments. The menu is divided into four sections: the top section is for moving between layers, the second section contains shortcuts for using Faders, the third section contains shortcuts for constructing environments and the bottom section contains utilities for updating environments and other Logic settings.

The **goto previous Layer** command is most useful in conjunction with the Edit menu's **Select Cable Destination** and **Select Cable Source** commands for tracing cables through multi-layered environment patches then returning to the starting layer. The **goto Layer of Object** command only functions from the All Objects layer. The All Objects layer is a list, in text format, of all objects on all layers in the environment. As with the **by Text** selection on the **View** menu, the objects are listed in the order of their unique IDs.

If you don't see an all objects layer on the layer menu, check the display section of Logic's preferences. It's toggled off and on there.

The faders section of the **Options** menu is handy both for instantly sending a group of Faders' values to your MIDI sound devices and for recording Fader snapshots in Logic. Since Logic records all Fader movements when in record mode, selecting the desired snapshot Faders with Logic in record-pause mode, then choosing **Send Selected Fader Values** will record the snapshot at the current song position. Recall that, the track on which the snapshot is recorded must eventually be cabled to an output port or back into the Faders for the snapshot to get to your MIDI sound devices. (The latter is necessary if you want the Faders to reflect the snapshot.)

In the environment shortcuts section, the **Define Custom Bank Messages...** choice was discussed under the *It's In The Bank* sub-heading of this section. The **Layer** sub-menu allows for creating and deleting layers. This is very handy when working on complex patches – create a new layer and temporarily move any irrelevant objects to it then move them back and delete the temporary layer when done.

If you copy an environment object or select **Define Template** from the **Apply Buffer Template to** sub-menu, Logic remembers the object's size, position, cable destination(s) and in the case of Faders, their In and Out definitions. If you select other objects of the same type and choose one of the selections from this sub-menu, the remembered parameters will be applied to the selected object(s).

In the case of Faders, the choices **Definition, channel increment** and **Definition, number increment** apply to both the In and Out definitions and the "increment" part means that the channel or -1- parameter will be incremented for each of the selected destination objects. The order of the objects for purposes of incrementing is their graphic order (left to right or top to bottom depending on whether they're "more" horizontal or vertical). Remember when using this function that the first of the selected destination objects matches the source definitions. If you want the incrementing to start with the first destination object, include the source object in the selection. For example, if you have four volume Faders and you want their channels to be successive, select the first Fader and use it to define the template then select all four Faders and use **Definition, channel increment**.

The **Clean up** options are fairly straight forward. The **Align Objects** command will line the objects up vertically or horizontally depending on whether they are "more" vertical or horizontal to start with. The spacing is determined by the objects' sizes and also by whether the outlets are displayed (no outlets results in closer spacing).

The **View** menu's **Snap Positions** option, when turned on (black dot), will force any object which is dragged or pasted to align to an invisible grid. The **Align Objects** function *does not* snap to this grid (control+dragging doesn't either). This can be a problem when a nicely aligned patch is dragged around or pasted to another layer. This forces all objects to snap to the grid and thus, fall out of alignment. One way to avoid this is to leave **Snap Positions** off. Another is to choose **Positions by Grid** after doing an alignment, then dragging objects as necessary to restore even spacing.

The last section of the **Options** menu is for updating the environment. The first choice, **Update OMS Equivalents**, creates a new layer named **OMS Objects** and creates appropriate environment objects corresponding (and mapped) to all the nodes in your OMS setup.

The second selection, **Import Settings...**, allows you to import Screen Sets, Transform Sets, Hyper Editor Sets, Score Instrument Sets, Score Style Sets and Score Settings from other Logic songs. Choosing this menu selection opens a dialog box for deciding which of these categories to import. An "X" in the corresponding button indicates that import is on.

The last selection, **Import Environment**, is used to import the environment from other Logic songs into the current song. Importing environments is discussed in detail under the next sub-heading, entitled:

6.3 Importing Environments from Other Logic Songs

With a little patching here and a little there, the environment quickly becomes quite complex. Since each Logic song has its own environment, automating the process of importing the environment from one song into another can mean trying to in some way match up two very complex and very different structures. As usual, Logic gives you a lot of choices and consequently, a lot to learn. Let's start with the simplest case: you have a patch in one song that you want to use in another.

Importing Single-Layer Patches

The easiest way to move a patch from one song to another is to either drag or copy & paste it. If the patch is small, open Environment windows in both songs and drag the patch from one to the other. If it is complex but all on one layer then select **Layer...** from the **Import Environment** sub menu of the **Options** menu. If the patch is on more than one layer, you can still copy & paste by moving the patch to one layer and this is almost always the best approach.

There is another consideration when copying patches between songs. If the patch contains "global" objects like Physical Input or to Sequencer, what happens to other patches in the destination song which are cabled to those objects? One solution is to avoid cabling directly to such objects when building a patch but rather, as we have done in the examples, start and end each patch with a neutral object like a Monitor or Transformer.

If a patch is self-contained and is not connected to any input or output ports, importing it will not disrupt the environment.

Importing environments obviously involves two songs – a source song and a destination song (*figure 78*). If you have two songs open, Logic will allow you to import from either song into the other. If only one song is open and you choose one of the importing options, Logic will consider the song in memory to be the destination song and will present an open-file dialog for you to choose the source song.

When the import menu is displayed, the line below the title bar will display which is the source and which is the destination song.

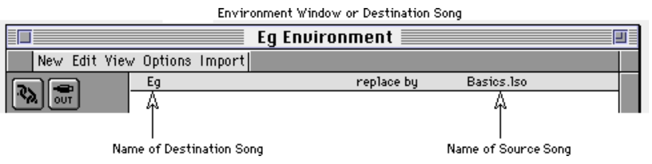


Fig. 78: Environment window with Import Options display enabled

Before getting into the **Import** menu, we need to mention a couple of things. When you create an object in the environment either from the

New menu, by option-dragging or with copy & paste, Logic assigns a unique ID to it. You can think of this ID as the lowest available ID number. Before you delete any objects, an object's ID is the same as its position in the list on the All Objects layer. Once you start deleting objects (thereby creating holes in the numbering), there is no way to tell an object's exact ID, but the All Objects layer still lists the objects in ID order.

An Object's ID never changes. It has nothing to do with the object's type, appearance, name or use.

One place the ordering of objects by ID becomes important is when loading a song into Logic. There is a checkbox in Logic's Song Preferences entitled **Send All Fader Values after loading**. (This changes from song to song as opposed to Logic's preferences which are the same for all songs.) If this Song Preference is checked and one Fader is cabled into another, the outcome after loading depends on the order of the Faders' ID numbers – the higher ID Fader value will be sent last and therefore may determine the setting of the lower ID Fader. (Note that SysEx Faders are never sent when a song is loaded.)

Logic provides five automatic import functions – the last four entries in the Import Environment sub-menu of the Options menu as well as the **Layer...** option described above.

Importing a Whole Environment

The Import Environment **Merge** option adds all objects from the source environment to the destination environment without disturbing any objects or cabling in the destination environment. Use this function when you have a song whose environment contains only the desired patch – e.g. songs whose whole purpose is to supply an environment but ensure these songs do not contain the unique objects like Physical Input, to Sequencer and Metronome.

Updating an Environment

The **Update** option is like merge except that objects with the same internal ID are replaced. Use this function with a growing Autoload song, for example, but be careful of deleted objects. Any object from the destination song which has been deleted in the source song may lead to erroneous replacements.

Updating Re: Port & Channel

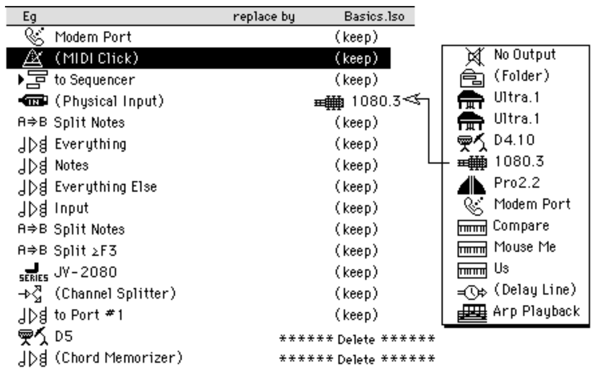
The **Replace by Port/MIDI Channel** option replaces all objects in the destination song which address a port and channel by an object in the source song (if any) which addresses the same port and channel. If you have imported a standard MIDI file using Logic's import function, and chosen **New** from the ensuing dialog, this option will allow you to import the portions of your Autoload environment necessary to play the MIDI file on your setup. There's a lot of guess work in Logic's automatic assignments, though. Using the **Import** menu is often a better choice.

Replacing the Whole Environment

The **Total Replace** option deletes everything in the destination song's environment and replaces it with the whole environment from the source song. You will generally be left with some serious re-instrumentation after you do this. But, if you have a song which you want to transfer to a whole new environment, this is often easier than moving all the sequences, folders, tempo changes, meter changes, key signature changes, markers, screen sets, etc. to an empty song with the desired environment.

Doing It All by Hand

In cases where none of the above options work, you can manually specify every detail of the import process. To do this you can select **Custom...** from the **Import Environment** sub-menu or if the source and destination songs are open, you can simply select the All Objects layer and activate the **Import** menu. The Environment window will show a list of all objects in the destination environment together with a list of actions for each object (*figure 79*).



Eg	replace by	Basics.lso
Modem Port		(keep)
(MIDI Click)		(keep)
to Sequencer		(keep)
(Physical Input)		1080.3
Split Notes		(keep)
Everything		(keep)
Notes		(keep)
Everything Else		(keep)
Input		(keep)
Split Notes		(keep)
Split ≥F3		(keep)
JV-2080		(keep)
(Channel Splitter)		(keep)
to Port #1		(keep)
D5		****** Delete ******
(Chord Memorizer)		****** Delete ******

Fig. 79: All Objects layer with import options enabled

If you click on an item in the right hand column, the familiar Instrument menu will appear. This is the Instrument menu from the source song, however, and from it you can choose what object is to replace the corresponding environment object in the left hand column.

In addition to their unique ID, there are other ways to identify environment objects. These include their name, their icon and the port & MIDI channel (Faders & Instruments only) they are assigned to. Logic allows you to match up objects for replacement by any or all of these criteria. If you don't want to replace an item manually or by one of the identification criterion, you can still choose to either keep or delete it.

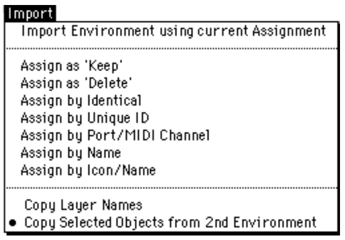


Fig. 80: The three sections of the manual import menu

The **Import** menu (*figure 80*) is in three sections. The top section causes the import to proceed as currently specified by the list. The middle section allows specific assignments to be made to the currently selected objects in the list. **Keep** and **Delete** mean just that. **Identical** means type, icon, name, port and MIDI channel. Otherwise, the current assignment is not changed. Likewise, **Unique ID**, **Port/MIDI Channel**, **Name** and **Icon/Name** will match the object on the left with an object from the source song if one exists and will leave the assignment unchanged, otherwise.

ℹ Any objects in the source environment which are reached by cables from imported objects are imported also.

If **Copy Layer Names** has "•" beside it, the source songs layer names will be imported, replacing the destination song's layer names.

Finally, if **Copy Selected Objects from 2nd Environment** has "•" beside it, any objects selected in the source environment and not copied as a result of an entry in the list will be copied to the destination environment.

Importing Multi-Layer Patches

This brings us to full circle to the easiest way to import a patch which occupies several layers. Go to each layer and select all objects. Go to the destination song and on the All Objects layer, ensure that all object replacements are set to **Keep**. (Select them all and use the **Import** menu's **Assign as 'Keep'** option if necessary.) Also ensure that **Copy Selected Objects from 2nd Environment** has "•" next to it. (Select it if not.) Then select **Import Environment using current Assignment**.

ℹ Objects are imported to the same layer number from which they came. Insert or delete layers in the destination song to have the objects imported where you want them (e.g. out of the way of existing patches).

6.4 So Much for The Basics — What's Next?

Whew! We've now covered all the environment objects and seen some simple applications for each one. We've also seen how to record the results of environment processing. And, we've looked at each of the environment's menus. Finally, we've examined how to move patches and even whole environments between songs.

This is where the fun begins. Having a basic knowledge of the environment, we can start using it to solve musical problems. In the *Environment Toolkit* you'll find:

- Tips and tricks for creating and using environments.

- A reference section with a complete description of each object and all of its parameters.
- Many small modules useful in creating custom environments.
- Logic songs containing easily imported environments including:
  - »MIDI Mixers
  - »Data switchers & routers
  - »LFO & Gate effects processors
  - »Time based MIDI effects processors
  - »Real time velocity and groove processors
  - »Analog-style sequencers
  - »Algorythmic composition tools
  - »Autoload Examples
  - Each environment includes detailed operating instructions as well as a complete description of how it is constructed.

Whether you want to create your own environments or simply make the best use of others, the *Environment Toolkit* will make the path easier. But, if you never use anything beyond what is covered in this beginner's guide, you still have incredible tools for customizing Logic.

Remember, five minutes and a few objects & cables can solve many musical problems that can't even be addressed by other sequencers.

Ordering The Environment Toolkit

The *Environment Toolkit* can be ordered directly from Len Sasso at:

Mail:	Swiftkick Productions P.O. Box 4257 Carmel, CA 93921
E-Mail:	LSasso@Swiftkick.com
Phone / FAX:	(408) 624-4123
Web:	www.swiftkick.com

Check out our web site for a complete description of *The Environment Toolkit* as well as constant updates on new environment features.

PostScript error (--nostringval--, --nostringval--)