Reflex

MIDI Implementation Details

Lexicon

Contents

Lexicon, Inc. • 3 Oak Park • Bedford MA 01730-1441 • Tel: 617 280-0300 • Fax: 617 280-0490

Lexicon Part No. 070-10748 Rev 1

Printed in U.S.A.

Reflex

MIDI Implementation Details

Introduction

Reflex MIDI implementation is designed to be compatible with the Lexicon LXP-1 and, thereby, all existing LXP-1 editor/librarian software, and the Lexicon MRC.

Definition of Terms

The following terms are used through out this document as defined here.

Active Programs or parameters which are currently in use are referred to as being **active**.

Nibblized The term **nibblized** is used to describe a method of transferring data blocks which are larger than the 7 bits allowed by MIDI. Nibblized data is broken up into significant "nibbles": the 16-bit value F32A hex would be nibblized as 0F 03 02 0A hex. As you can see, the high nibble is forced to zero and low nibble contains the actual data. Although not terribly efficient, this is fairly straightforward to implement and to interpret when viewing raw hex data.

Packed The term **packed** is used to describe another method of transferring data blocks which are larger than the 7 bits allowed by MIDI. Packing the data is an extremely efficient method of transferring different size data types via MIDI, although it is more difficult to implement and to interpret. Packing basically strips the MSB off of each byte (8 bits) of data and assembles them in an additional byte. It therefore takes 8 MIDI bytes to transmit 7 raw data bytes. See *Data Packing* for a detailed description.

Parameter A **parameter** is an attribute of the system that changes when front panel controls are altered, or when the system receives MIDI SysEx messages. See *Parameters* for a complete description of parameters.

Preset A **preset** is a setup that can *not* be changed by the operator. Presets are stored in ROM (read only memory) and are typically used as a starting point in the creation of registers. Note that when presets and registers are selected they are copied into "working" setups that *can* be edited, then stored in registers. The operator cannot overwrite the system's presets.

Program, Algorithm **Program** and **Algorithm** refer to a microcode program loaded into the Lexichip to produce a specific type of audio effect. Reflex contains 8 microcode programs (listed and described in the appendix).

Register A **register** is a setup that can be changed by the operator and stored in nonvolatile memory within the system. The system supports 128 user registers, numbered 0-127 (1-128 on the front panel display).

Setup A **Setup** is a group of changeable attributes that define how the system will operate. In software, a setup exists as a table containing values for most of the system's attributes. Setups can be stored in, or loaded from, the system's nonvolatile EEPROM and/or transferred in and out of the system via SysEx dumps. See *Data Dumps* for a byte-by-byte definition of a setup

Parameters

All of the System Exclusive activity involving parameters requires a parameter number and a data value. Each parameter changes different aspects of how the system is currently working. There are two major classifications of parameters: **System** parameters and **Setup** parameters.

System Parameter

There is only one system parameter: Setup number. The value of Setup number, parameter **64** (40 hex), controls which setup (register/preset) is running. Setups 0-127 refer to registers 1-128, while setups 128-144 refer to presets 1-16. For example:

byte	1	2	3	4	5	6	7	8	9	10
	F0	06	02	50	40	00	00	0**3**	0**B**	F7

This is a nibblized SysEx parameter change message (type 5) that changes the current setup to register number 3B hex (59 decimal).

Setup Parameters

Setup parameters are defined as all of the parameters stored in a setup (user register or preset). In general, these parameters effect how a setup sounds or is used by the system. There are four categories of setup parameters: Algorithm Number, Audio, Setup Name and MIDI Patch. These categories are listed below with the range of parameter numbers with which they can be accessed.

Category	Parameter Numbers
Algorithm Number	65 (41 hex)
Audio	0-10 (0-A hex)
Setup Name	32-47 (20-2F hex)
MIDI Patch	48-63 (30-3F hex)

Note that parameters 60-63 are *not* stored with user registers.

Algorithm Number

(LXP-1 documentation refers to Algorithm Number,or Program ID, as a *System Parameter*, but it is more accurately described as a setup parameter as it identifies the DSP program run with a given setup and is stored with both preset and user register setups.)

The most important setup parameter is Algorithm Number. Accessed using parameter **65** (41 hex), the Algorithm Number defines one of the following DSP algorithms to be used with a given setup:

Name (MRC)	Algorithm Number	Description	Used with Preset:
Reverb	1	Rooms and Halls	1-6
Plates	2	Plate Emulation	9-11
Chorus 1	3	Flanger	12
Delay 2	4	Multi-Tap Delay	15
Chorus 2	5	Resonator	16
Inverse	6	Inverse Room	7
Gate	7	Gated Reverb	8
Delay 1	8	Chorus/Delays	13, 14

The Algorithm Number is stored in a setup as an 8-bit value. When changed via a SysEx message, a full 16-bit value should be transmitted with the high bits all set to zero. For example:

byte	1	2	3	4	5	6	7	8	9	10
	F0	06	02	50	41	00	00	0**0**	0**8**	F7

This is a nibblized SysEx parameter change message (type 5) that changes the current algorithm to number 8. Note that the two high nibbles (in bytes 6 and 7) are set for zero. Remember that the Algorithm number is one of the parameters of a given setup. Changing the Algorithm number is the same as changing any of the other parameters except that the values of the Audio Parameters may produce dramatically different effects with the new algorithm than the old. Some caution should be exercised when changing only the algorithm number as some Audio Parameter values which are legal for one algorithm may not be legal for another. The values are checked and limited before they are applied to the DSP but they may appear as being out-of-range when viewed on an MRC or via Reflex's Advanced Programming Mode (APM). Reflex will ignore new algorithm number values if they are not between 1 and 8.

Audio Parameters

Audio parameters directly effect the sound of an algorithm in a given setup. Because, in conjunction with the Algorithm Number, they *characterize* the sound of a setup, they generally have different values for each setup that uses the same algorithm. In this way, a single algorithm can produce dramatically different sounding effects. While there are always 11 of these additional parameters for every setup, the actual effect they have on each algorithm varies dramatically. See the Appendix for a complete listing of Audio Parameter numbers and their effect in each algorithm.

The following table outlines the Audio parameters:

Parameter #	Description
0	Front Panel DECAY Parameter (actual effect varies)
1	Front Panel DELAY Parameter (actual effect varies)
2	Front Panel FX LVL Parameter (effect output level)
3	Different for each algorithm
4	Different for each algorithm
5	Different for each algorithm
6	Different for each algorithm
7	Different for each algorithm
8	Different for each algorithm
9	Different for each algorithm

*10 (A hex) Effect Input Level (used for BYPASS)
*Note that this parameter is not available in Reflex's APM.

Each of these parameters is stored as a 16-bit value in a setup.

Audio parameters can be broken down into two basic categories: Bipolar (+ and -) and Unipolar.

> **Bipolar** parameters typically consist of values ranging from 0x4000 (most negative) to 0xBFFF (most positive) with 0x8000 representing a value of zero.
>
> **Unipolar** parameters typically range from 0x8000 (min) to 0xBFFF (max). *There are exceptions.* Refer to the Appendix for a complete listing of all algorithms with their associated Audio Parameters including the range of legal values for each.

Please note that many of the parameters lack the resolution implied by this range. Software written to control these parameters should be sensitive to the finest effective resolution which is indicated in the Appendix. The range of values available for each parameter demonstrates a sensitivity of resolution by design. The front panel parameter controls on the LXP-1 have a fixed resolution of 16 positions (or 4 bits).

In order to allow all of the parameters to be cleanly mapped to these encoders, the most significant bits of each parameter are shifted to the right so that any of the parameters can be coarsely adjusted via the same 4 bits. 8000 hex was chosen as a zero point to further simplify coding.

Note that when a new parameter value is sent to a Reflex via SysEx, the previous value of that parameter in the active setup is overwritten. Similarly, changing parameter values via the front panel overwrites the values set via SysEx.

Parameter 10 (Input Level) is a bit different from the other Audio parameters in that its value is not stored in user registers. This parameter is always set to maximum (0xBFFF) when a new preset or register setup is loaded unless the system is in Bypass, in which case the parameter is left at 0x8000 (min). In many respects, parameter 10 could be considered a system parameter. Although you can change the value of parameter 10 via SysEx, the system software will assume that the parameter is set where last left (Bypassed or not) — so proceed with caution. This gives you the ability to produce a bypass that mutes the output, in addition to the input via MIDI, if desired.

Setup Name Parameters

Setup Name parameters are actually just a string of 16 8-bit memory locations set aside in each setup to store a string of text describing the setup. Each “parameter” simply provides access to one of the character locations. Though not required, it is suggested that the string be null terminated if possible.

You must access each character one at a time to change or read the name of the active setup. The parameters are numbered **32-47** (20-2F hex).

Note that the MRC will only display the first 8 characters of the name, even though 16 are available. Reflex presets are stored with the name parameters: Preset1, Preset2, etc. Also note that the MRC V3 software does *not* transmit these parameters as part of an active setup dump. (MRC V4 does transmit them as part of an active setup dump.)

MIDI Patch Parameters

MIDI patch parameters allow generic MIDI controller and note information to dynamically control a given parameter. Reflex supports four MIDI patches per setup, which can be stored in a user register. Each MIDI patch consists of:

a source (parameters **48-51** dec, 30-33 hex)
a destination (parameters **52-55** dec, 34-37 hex)
a scale value (parameters **56-59** dec, 38-3B hex)
an offset* (parameter **60-63** dec, 3C-3F hex)

* This parameter is not stored in user registers.

See *MIDI Patching* for more detailed information

MIDI Channel Selection

The MIDI channel to which a given Reflex system will respond to MIDI messages can be set in one of two ways: from a menu item in the Advanced Programming Mode (APM) or by “learning” the channel from incoming MIDI data.

In APM, set the REGISTER/PRESET encoder to position 12 and turn the VALUE encoder to select the desired MIDI channel.

To learn a MIDI channel, press and hold the PARAMETER/LEARN button until the scaling (see *MIDI Patching*) value appears then, still holding down PARAMETER/LEARN, send the unit a MIDI message containing channel number information. The button may then be released. Messages received while the button is held down will be honored regardless of channel number and the unit will adopt the transmitted channel number as the operational MIDI channel. System Common or Running Status messages will not cause the unit to change channels since these messages do not contain MIDI channel information. SysEx messages in Reflex protocol will work since they *do* contain channel information (Reflex considers MIDI channel its Device ID.)

Note that you may inadvertently clear a patch if you attempt to “learn” the MIDI channel of incoming data while a parameter with a patch assigned to it is selected. Refer to *MIDI Patching* for additional information.

System Exclusive

Reflex system software was written to support the same System Exclusive message protocols established for the LXP-1. The following sections should provide all of the information necessary to write software to interface with Reflex using System Exclusive.

Types of SysEx Messages

To understand the types of SysEx messages used with Reflex you need to know what SysEx is used for by the system. In general, SysEx messages are used to:

1. Move setup data out of the system (dumps)
2. Move setup data into the system (loads)
3. Change system parameters
4. Execute system tasks
5. Request information from the system

These tasks can be further broken down into particular cases: (Note the "types" indicated refer to message types as defined in the *Message Protocols*.)

1. Setup data can be moved out of the system with the following types of SysEx messages:
   - Active Setup Dumps (type 0)
   - Stored Setup Dumps (single registers) (type 1)
   - All Register Dumps (type 4)

2. Setup data can be moved into the system with the following types of SysEx messages:
   - Active Setup Loads (type 0)
   - Inactive Register Loads (single) (type 1)
   - All Register Loads (type 4)

3. Parameters can be changed using the following types of SysEx messages:
   - Packed Parameter Adjust (type 2)
   - Nibblized Parameter Adjust (type 5)

4. The following system tasks can be performed via SysEx messages (type 6):
   - Store the current setup in a register
   - Recall a stored register (stored in the Reflex)
   - Set Bypass mode

5. The following information can be requested of the system via SysEx: (type 3)
   - Active setup data
   - Contents of a single inactive register
   - Contents of all registers (dump all)
   - Packed parameter data
   - Nibblized parameter data

SysEx Message Protocols

Reflex supports the following SysEx data message types:

0 Active Setup Data
1 Stored Register Data (single)
2 Packed Parameter Adjust
3 Requests
4 All Registers Data
5 Nibblized Parameter Adjust
6 System Tasks

The following sections describe how to construct SysEx messages for Reflex and how SysEx messages are used/interpreted by the system.

0 Active Setup Data

Active Setup Data can be sent to a Reflex system or, if requested, transmitted from a Reflex. When sent to a Reflex, Active Setup Data is loaded into the current working setup register or preset (depending on which mode the system is currently in). When transmitted from Reflex in response to a request, the current working preset or register setup is transmitted. In either case, the data is transported in the following format:

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**0**n	0000 nnnn	0 = the message type and n = midi channel: 0-F (0-15)
5	38	0011 1000	Packed data byte count =56 (49 unpacked)
6	–	0vvv vvvv	Setup data* in 8/7 packed format. v = data, MSB must be 0
.			
.			
.			
61	–	–	0vvv vvvv
62	–	–	0sss ssss Checksum** of data bytes (done on the bytes in packed format)
63	F7	1111 0111	End of sysex message

* See *Data Dumps* "for more information.
** Checksum is calculated by adding all of the data bytes together and using the 7 lowest bits of the sum.

Reflex displays "dC" (**d**ump **c**urrent setup) when an Active Setup Data dump out of the system is occurring. "LC" (**l**oad **c**urrent setup) is displayed when an Active Setup Data dump into the system is in process. The plus sign flashes during either type of dump.

Note: This is the way that the Reflex updates the MRC V4 when the operator presses the <ENTER> key. The MRC transmits a "request (type 3) message for active setup data and the Reflex responds with this data.

1 Stored Setup Data

This data type is used to transfer a single register setup into and out of a Reflex. This command essentially works in the background with one of the user registers, leaving the active setup unaffected.

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**1**n	0001 nnnn	1=message type, n=midi channel: 0-F (0-15)
5	–	0ppp pppp	p=register number 0-127
6	38	0011 1000	packed data byte count (56 decimal)
7	–	0vvv vvvv	Setup data* in 8/7 packed format
.			
.			
.			
62	–	0vvv vvvv	
63	–	0sss ssss	checksum** of data bytes (sum done on the bytes in packed format)
64	F7	1111 0111	End of SysEx message

* See *Data Dumps* .
** Checksum is calculated by adding all of the data bytes together and using the 7 lowest bits of the sum.

Reflex displays “dS” (**d**ump **s**tored setup) when a Stored Setup Data dump out of the system is occurring. “LS” (**l**oad **s**tored setup) is displayed when a Stored Setup Data dump into the system is in process. The plus sign flashes during either type of dump.

If more than one register dump is sent to the system within a 1 second timeout, the system waits for any additional setups, then stores all of the stored setups in the system’s non-volatile EEPROM. A chase pattern is displayed on the front panel while the data is copied to the EEPROM. During this time (approximately 14 seconds) all incoming MIDI data, operator button presses and encoder turns will be ignored.

2 Packed Parameter Adjust

This type of message is used to change a parameter in the active setup — the effect should be heard immediately. See *Data Packing* for a description of data packing, and *Parameter Data* for more information about the parameters.

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**2**n	0010 nnnn	2=message type, n=midi channel: 0-F (0-15)
5	–	0ppp pppp	p=parameter number 0-127
6	–	0vvv vvvv	data in 8/7 packed format (Section 3,4)
7	–	0vvv vvvv	
8	–	0vvv vvvv	
9	F7	1111 0111	End of SysEx message

This is the SysEx format that is output from Reflex when the operator moves a front panel encoder on the front panel.

Examples:

byte	1	2	3	4	5	6	7	8	9	
	F0	06	02	20	**00**	0**2**	**00**	**04**	F7	Instructs the Reflex on MIDI channel 1 to change the value of parameter 1 to 8004 hex.
	F0	06	02	2**5**	**40**	00	**0B**	00	F7	Instructs the Reflex on MIDI channel 6 to load register 12 (B hex=11, + 1 because actual numbers start at 0)

The effect of incoming SysEx parameter changes can be observed on the Reflex front panel in several ways. In the normal (not APM) operating mode, the system will revert the “selected” parameter number to the incoming parameter if it is parameter 0-2 and display the parameter value for 3 seconds. If other parameters come in this mode, all three parameter LEDs flicker to indicate that the parameter value can be viewed in Advanced Programming Mode.

In Advanced Programming Mode, you can view the changes you are making to a parameter by turning the REGISTER/PRESET encoder to the appropriate parameter number. All three parameter LEDs will flash to indicate that incoming MIDI has altered this parameter.

3 Requests

This type of message allows the operator to request information from the receiving Reflex without actually touching the unit. The five types of requests (request code) are typically used by editor and/or librarian programs to extract data from a Reflex system.

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**3**n	0011 nnnn	3 = the message type and n = midi channel: 0-F (0-15)
5	–	0ccc cccc	c = request code: 60 hex=Active Setup Data 61 hex=One Register 62 hex=Packed Param Data 64 hex=All Registers Data 65 hex=Nibble Parameter Data
6	–	0ppp pppp	p=register number 0-127 for c (above) =61 p=parameter number for c=62 or c=65 else a value is present but ignored
7	F7		End of SysEx message

Examples:

byte	1	2	3	4	5	6	7	
	F0	06	02	30	60	00	F7	Instructs the Reflex on MIDI channel 1 to transmit an Active Setup Data type dump of the active setup.
	F0	06	02	3**2**	**61**	0**5**	F7	Instructs the Reflex on MIDI channel 3 to transmit a Stored Register Data type dump of register 6.

4 All Registers Data

This type of message allows the contents of all 128 register setups to be copied into and out of the system. When written to a Reflex, the previous contents of the user registers are overwritten. Note that because of the amount of data that gets transferred, this operation typically takes about 10 seconds to complete.

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**4**n	0100 nnnn	4=message type, n=midi channel: 0-F (0-15)
5	38	0011 1000	Hi byte of number of packed data bytes (see below)
6	00	0000 0000	Lo byte of number of packed data bytes
7	–	0vvv vvvv	Data* in 8/7 packed format
.			
.			
.			
7174	–	0vvv vvvv	
7175	–	0sss ssss	Checksum** of data bytes (sum done on the bytes in packed
7176	F7	1111 0111	End of SysEx message

* See *Data Dumps* for information about this data.
** Checksum is calculated by adding all of the data bytes together and using the 7 lowest bits of the sum.

The number of packed data bytes is calculated as follows:

number of bytes per register=56
number of registers=128

56* 128=7168=x1c00=3800 in 7-bit (shift to the left 1 bit)

Reflex displays “dA” (**d**ump **a**ll setup) when an All Registers Data dump out of the system is occurring. “LA” (**l**oad **a**ll setup) is displayed when an All Registers Data dump into the system is in process. The plus sign flashes during either type of dump.

When an All Registers Data dump is sent to Reflex the data is copied into SRAM, then copied into the system’s non-volatile EEPROM. A chase pattern is displayed on the front panel during the EEPROM write and all incoming MIDI data, operator button presses and encoder turns are ignored (approximately 14 seconds).

5 Nibblized Parameter Adjust

Nibblized Parameter Adjust allows you to change the value of any parameter in the active setup. This is probably the most straight forward way of changing parameters via MIDI SysEx and, considering that the entire message is only one byte bigger than the packed version, it is also the preferred method. (This is how the MRC sends parameter data to Reflex.) See *Parameters* for additional information.

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**5**n	0101 nnnn	5=message type, n=midi channel: 0-F (0-15)
5	–	0ppp pppp	p=parameter number 0-127
6	0d	0000 dddd	(hi) d=16-bit data sent in nibbles
7	0d	0000 dddd	
8	0d	0000 dddd	
9	0d	0000 dddd	(lo)
10	F7	1111 0111	End of SysEx message

Examples:

byte	1	2	3	4	5	6	7	8	9	10	
	F0	06	02	5**0**	0**2**	0**8**	0**0**	0**0**	0**0**	F7	Instructs the Reflex on MIDI channel 1 to set parameter 3 to 8000 hex.
byte	1	2	3	4	5	6	7	8	9	10	
	F0	06	02	5**F**	0**5**	0**B**	0**F**	0**C**	0**0**	F7	Instructs the Reflex on MIDI channel 16 to set parameter 6 to BFC0 hex.

6 System Tasks

This type of message allows you to instruct Reflex to perform various system level activities including: storing the active setup in a register, recalling a register (making it active) and setting the Bypass mode.

Byte#	Value Hex	Binary	Description
1	F0	1111 0000	System Exclusive
2	06	0000 0110	Lexicon ID
3	02	0000 0010	LXP-1 ID
4	**6**n	0110 nnnn	6=message type, n=midi channel: 0-F (0-15)
5	–	0eee eeee	e =event code: 70 hex = store register 71 hex = recall register 72 hex = set bypass mode
6	–	0ppp pppp	p=argument
7	F7	1111 0111	End of SysEx message

For “store” and “recall” operations, the argument is the register number (0-127). For “bypass”, the argument turns Bypass on or off: 0=OFF, 1=ON.

Example:

byte	1	2	3	4	5	6	7	
	F0	06	02	6**0**	**70**	**03**	F7	Instructs the Reflex on MIDI channel 1 to store the active setup in register 4
	F0	06	02	6**3**	**71**	**09**	F7	Instructs the Reflex on MIDI channel 4 to recall register 10
	F0	06	02	6**0**	**72**	**01**	F7	Instructs the Reflex on MIDI channel 1 to go into bypass mode

SysEx Errors

When a dump of SysEx data is received by Reflex, the system checks the number of bytes received and the checksum of the data. If either of these is incorrect, or if a message is started but does not finish, Reflex reports the error by alternately flashing the letters “Er” and an error code number. The error codes are as follows:

1 Got wrong checksum

2 Got wrong number of bytes

3 Timed out waiting for message

The error message is displayed until the operator touches a button or encoder.

SysEx Output in Response to Front Panel Activity

When a front panel encoder is turned, Reflex automatically transmits a “packed parameter adjust” message reflecting the new parameter value. The transmitted parameter number is 0 for the VALUE encoder if the selected parameter is DECAY, 1 if the selected parameter is DELAY, 2 if the selected parameter is FX LVL. When the REGISTER/PRESET encoder is turned, parameter number 64 (40 hex) is transmitted.

Note that Reflex has only one MIDI “output” jack which can be configured as a true MIDI OUT or a MIDI THRU depending on the position of an internal jumper. The default position of this jumper when Reflex is shipped from the factory is MIDI OUT. If no MIDI data is being observed at the MIDI OUT jack when front panel knobs are turned, the jumper is probably set for THRU.

MIDI Patching

The MIDI patching system was developed to allow Reflex’s Audio Parameters to be altered using generic MIDI events. The Reflex system supports 4 MIDI patches per setup which can be stored in a user register. Patches can be created from the Reflex front panel (with a MIDI controller connected) or externally as part of a setup. When created from the front panel, the operator follows a simple programming sequence as described later in this section, then stores the setup in one of the user registers for future use. To create a setup which includes patches outside of Reflex (in an editor, for example), you need to understand the components of a patch and how they are derived.

Components of a Patch

There are, effectively, four elements that contribute to a patch: a source, a destination, a scale and an offset. The source is the type of generic MIDI event that will be used in the patch. The destination is the audio parameter which will be modified by the patch. Scale refers to a value which defines the mathematical relationship between the MIDI data coming in from the specified source to the audio parameter defined as the destination. The offset is a value that is calculated internally by Reflex. The offset value, which is based on the scale value and the current source data value, is added to the internally stored parameter value enroute to the DSP. Although the offset is not part of the setup sent to Reflex, it can be read back from a Reflex to determine the effect of a given patch (See *Data Dumps* for a byte-by-byte description of a setup, and *Parameter Map* in the Appendix for parameter numbers of the offsets.)

Reflex responds to five types of generic MIDI messages:

Source Number	Message Type
0-63	MIDI Continuous controller 0-31, 64-95
64	Note On (last note played)
65	Last note’s velocity (only programmable via an external editor)
66	Channel aftertouch value
67	Pitch bend value

The Source Numbers in the table represent legal numbers that you can enter as source values when creating patches on setups outside of the Reflex. The numbers are really just indices used by the system software to keep track of source data. When created within a Reflex, this tracking is handled by the system.

Once defined as a source, the “data” sent with these MIDI messages is multiplied by the scale value, and then multiplied by 2 to generate an offset value. This offset is added to the current value of the destination parameter to change the audible effect. With a patch defined for a given parameter, the current value of the parameter effectively becomes a “base value” for the patch.

Scaling

Scaling refers to a value which defines the mathematical relationship between the source and the destination. In Reflex, this relationship is defined in positive and negative percentages. Scale can be in the range of +199% to -199%. A scale of +100% would directly map a MIDI source with a range of 0-127 to a Reflex parameter with a range of 1-128. A scale of +50% would map a MIDI source with a range of 0-127 to one half the range of a Reflex parameter. The actual range of parameter values is based on the "base value" of the parameter. In the previous example, a base value of 1 would yield control of parameter values 1-64. A base value of 64 would yield control of parameter values 64-128. A scale of +199% would map MIDI source values of 0-64 to parameter values of 1-128, etc...

Negative percentages produce similar source/destination value relationships, but with inverse response. A MIDI control range of 0-127 would produce a parameter value range of 128-1 with a scale percentage of -100%. The use of percentages greater than 100% is most useful with MIDI sources of limited range, and with bipolar parameters.

The actual scale value stored with a setup is a single byte, meaning there are really only 256 discrete scale values. The user interface scales the "scale" value to give the operator the illusion of 398 percentage steps. This shows up in operation as effective difference between certain percentage steps and accuracy of the displayed percentage. The following table illustrates the relationship between the value of the scale byte in a setup to the percentage:

%	byte value
0	0x00
+50	0x20
+100	0x40
+199	0x7F
-199	0x80
-100	0xC0
-50	0xE0

In Reflex, the displayed 8-bit scale percentage which is stored as a value of 0-127, is calculated using a 16-bit scratch scale value to give a + and - 0-199 range as follows (in "C"):

```
BYTE sign_mode;
UWORD displayed_value = scale_scratch;

if (displayed_value > 0x7fff) /* mid value */
{
    sign_mode = MINUS_SIGN;
    working_value = 0 working_value;
}
else /* must be < MID_VALUE */
{
    sign_mode = PLUS_SIGN;
}

displayed_value = displayed_value    /* 1      */
        + (displayed_value >> 1)     /* +1/2   */
        + (displayed_value >> 4);    /* +1/16  */
displayed_value = displayed_value >> 8;

if (displayed_value != 0)
    set_sign_display(sign_mode);
else
    turn_sign_off();

display_number(displayed_value);
```

Offsets

The patch offsets are the actual values that get added to the base parameter values en route to the DSP. Offsets are continuously recalculated by Reflex by multiplying the patch source value and the scale value times 2 as a background task. This offset is added to the stored base value of the specified destination parameter. This sum of the offset and base value is the value sent to the DSP chip. The programmer should be aware that because the offset values are continually updated internally, any changes you make to the offset value externally will be internally overwritten shortly thereafter. For this reason, offset values are not stored when registers are saved. When a new setup is recalled, new patch offsets are realculated. The offset parameters are most useful to the programmer when read from the Reflex to determine the MIDI patch contribution to the parameter values transmitted by the Reflex.

Miscellaneous Notes

1. Any patches that are not made/defined should have the source and destination parameters filled with 7F hex which is defined by the system as a NULL PATCH. However, any values greater than or equal to 7F hex will be recognized by the system software as NULL as well.
2. When Reflex receives a parameter change (via SysEx or front panel), the received value is used as the new base parameter value if a patch has been made. When Reflex transmits a parameter value, however, the value transmitted is the base parameter value *plus* the MIDI patch offset.
3. Negative scaling percentages tend to run high due to the variety of parameter ranges in the system. Full inverse control of a parameter can usually be achieved with percentages less than 99%.
4. Some parameters and patches appear in "*unused*" parameter slots in Reflex's APM mode due to the elimination of duplicate parameters and their subsequent remapping. Refer to *Parameter Map* in the Appendix for additional information.

Creating a Patch from the Normal Operating Mode

To create a patch from the normal operating mode:

1. Select a parameter with the front panel PARAMETER/LEARN button. When the target parameter is reached, the operator must wait until the display returns to the preset/register display (parameter LED stops blinking).
2. Press and hold the PARAMETER/LEARN button until the display changes to the "scaling" mode (about 1 second).
3. Still holding down PARAMETER/LEARN, send the system a sampling of MIDI data by wiggling the control source.
4. Set the scale value by turning the VALUE encoder.

The patch is locked in on release of the PARAMETER/LEARN button. Fine tune the patch by adjusting the VALUE encoder for the parameter which becomes the base value of the patch.

Creating a Patch from Advanced Programming Mode

To create a patch from the Advanced Programming Mode (APM):

1. Select a parameter with the REGISTER/PRESET encoder.
2. Press and hold the PARAMETER/LEARN button until the display changes to the "scaling" mode (about 1 second).
3. Still holding down PARAMETER/LEARN, send the system a sampling of MIDI data by wiggling the control source.
4. Set the scale value by turning the VALUE encoder.

Clearing a Patch

In the normal operating mode, there is only one method of clearing patches: select the parameter with the patch then press and hold the PARAMETER/LEARN button until the scaling value appears. If the button is released with no MIDI values sent to Reflex, the patch is cleared. The letters “CL” are displayed for approximately 1 second to indicate that the patch was cleared.

In APM there are two ways to clear a patch. The first method is similar to the normal mode: select the parameter, press and hold the PARAMETER/LEARN button until the scaling value appears, then release the button without sending MIDI to Reflex. The second method is to select the parameter with the patch to be cleared and press the STORE/CLEAR button. With either method, the letters “CL” are displayed for approximately 1 second to indicate that the patch was cleared.

Note that you may inadvertently clear a patch if you attempt to “learn” the MIDI channel of incoming data if the selected parameter has a patch assigned to it.

Displayed Parameter Value

Once a patch has been made to a parameter, the displayed value of that parameter no longer represents the value that will be stored in a setup. The value that appears on the display is the effective parameter value derived from the stored value and the offset added by the patch. With a patch made to a parameter, the stored parameter value becomes the base value of the patch. Once the patch is made, the base value only appears when the operator turns the VALUE encoder. The base value is displayed for approximately 3 seconds before reverting to the effective parameter value (in APM) or the setup number in the normal operating mode. The effect that the patch has on the parameter can be observed by selecting the parameter.

In the normal operating mode, if patches are made to one of the three available parameters, and MIDI control data comes in for that patch, the appropriate parameter LED will light, and the value will be displayed for about 3 seconds. MIDI data for patched parameters not available from the normal operating mode will be indicated by flashing all three parameter LEDs.

Multiple Patches to a Single Parameter

The patching system in Reflex supports the patching of multiple MIDI control sources to a single parameter. This is implemented by simply repeating the patching sequence for each of the desired control sources for a maximum of 4 patches. The number of patches to a given parameter can be determined in the Advanced Programming Mode by counting consecutive “quick” flashes of the decimal point when the parameter is selected. This is also a useful indicator of whether or not patches are made to a parameter at all. If no patches are made to the parameter, the decimal point will stay off.

To determine the number of patches via MIDI, count the number of sources and/or destination bytes that are not 7F hex in the active setup.

MIDI Clock

Two of the Reflex algorithms respond to incoming MIDI clock messages by recalculating their delay times to musically related values. When either the “Multi-Echoes” (A4) or “Chorus (A8) algorithm is active, and MIDI clocks are being fed to the system, delay lengths are automatically calculated to match a defined “Echo Rhythm” at the tempo of the incoming MIDI clock. The Echo Rhythm is set via parameter 10 in Advanced Programming Mode (9 via SysEx) per the following table:

1	64th
2	Thirty-second
3	Sixteenth Triplet
4	Sixteenth note
5	Eight Triplet
6	Dotted Sixteenth note
7	Eighth note
8	Quarter Triplet
9	Dotted Eighth note
10	Quarter note
11	Half triplet
12	Dotted Quarter note
13	Half note
14	Whole note

When a change is detected in the MIDI clock tempo, Reflex recalculates the delay times accordingly. If the selected Echo Rhythm cannot be produced with the available audio memory, the system divides the interval in half in an attempt to keep the delay times musical. For instance, if the CANYON preset is loaded, Echo Rhythm is set for whole note (14) and the tempo of the incoming MIDI is 120 BPM, Reflex will set the delays for a half-note. This is because there is insufficient memory in the system to produce whole-note echoes at this tempo. This happens even more frequently in the Multi-Echoes algorithm because there are four delay taps to be set.

Note that this is intended to be used as a real-time control tool and, as such, the delay times generated by the MIDI clock are *not* saved in user registers. The Echo Rhythm, however, is. Also note that any value of Diffusion greater than 1 in these algorithms will cause a 30ms offset in delay values against the MIDI clock.

For the most rhythmically accurate delays when slaving to MIDI clock, set all of the delay parameters to 1. This is particularly important for the secondary delays as the delay setting there will be added to the delay calculated from the MIDI clock producing delay times that do not appear to be synchronized to the MIDI clock.

MIDI Bypass

The Reflex software was written to allow one of several types of MIDI message to be used to toggle the system in and out of bypass. The system can be programmed to toggle bypass in response to any of the following types of MIDI messages:

Program change
Continuous controller
Note ON

When shipped from the factory, MIDI bypass is set for Program Change, program number FF hex which is an illegal Program Change number. This is done to effectively disable MIDI bypass. In fact, you can still toggle the bypass via MIDI at this program number using a SysEx parameter change message, parameter 64. (See *Parameters.*)

MIDI bypass is programmed to other MIDI messages by putting the system in bypass using a footswitch, pressing and holding the PARAMETER/LEARN button and sending the system the MIDI message you want it to use for bypass. Program Change and Note messages toggle the bypass mode each time they are received. Continuous controllers toggle the bypass mode each time the value passes from less than 64 to greater than 64.

It is possible to inadvertently clear a MIDI Patch while programming MIDI bypass. To avoid this, select a parameter which does not have a patch assigned, or reload your setup after learning MIDI bypass.

MIDI bypass can be disabled by putting the system in bypass, pressing the LEARN button and toggling the system out of the bypass mode via the footswitch.

Please note that the bypass mode of the system can also be set using a “System Tasks” SysEx message.

Data Packing

Data Packing is a method of transferring data types larger than 7 bits over MIDI (which requires raw data to be less than 8 bits). The alternative to Data Packing is passing data in the Nibblized format, which is fine for small packets of data, but inefficient for large blocks.

The basic concept behind the packing algorithm is to take the MSB of 7 bytes of data and collect them into an 8th byte. In this way we can pass the 7 bytes with the MSB set to zero, respecting the MIDI requirement, with the extra byte containing the MSBs of the 7 bytes. Confusing but effective.

Unpacked data:

bits	7	6	5	4	3	2	1	0
byte 0:	**a7**	a6	a5	a4	a3	a2	a1	a0
byte 1:	**b7**	b6	b5	b4	b3	b2	b1	b0
byte 2:	**c7**	c6	c5	c4	c3	c2	c1	c0
byte 3:	**d7**	d6	d5	d4	d3	d2	d1	d0
byte 4:	**e7**	e6	e5	e4	e3	e2	e1	e0
byte 5:	**f7**	f6	f5	f4	f3	f2	f1	f0
byte 6:	**g7**	g6	g5	g4	g3	g2	g1	g0

Packed output:

bits	7	6	5	4	3	2	1	0	
byte 0:	0	**g7**	**f7**	**e7**	**d7**	**c7**	**b7**	**a7**	(MSBs)
byte 1:	0	a6	a5	a4	a3	a2	a1	a0	
byte 2:	0	b6	b5	b4	b3	b2	b1	b0	
byte 3:	0	c6	c5	c4	c3	c2	c1	c0	
byte 4:	0	d6	d5	d4	d3	d2	d1	d0	
byte 5:	0	e6	e5	e4	e3	e2	e1	e0	
byte 6:	0	f6	f5	f4	f3	f2	f1	f0	
byte 7:	0	g6	g5	g4	g3	g2	g1	g0	

Large amounts of data are handled by sending multiple blocks of packed data until all data is transmitted (least significant data sent first). If it is necessary to send a packet of less than seven bytes, only those bytes are sent, preceded by the MSB byte with the active MSBs right-justified. For example, to send a 16-bit word, the format is as follows:

Unpacked data:

```
byte 0: a7 a6  a5  a4 a3 a2 a1 a0 (least significant)
byte 1: b7 b6  b5  b4 b3 b2 b1 b0
byte 2: c7 c6  c5  c4 c3 c2 c1 c0 (most significant)
```

Packed output:

```
byte 0: 0   0   0  0  0  c7 b7 a7
byte 1: 0   a6  a5 a4 a3 a2 a1 a0
byte 2: 0   b6  b5 b4 b3 b2 b1 b0
byte 3: 0   c6  c5 c4 c3 c2 c1 c0
```

For example:

Unpacked data:

```
byte 0: 1   0   1  1  0  1  1  1
byte 1: 0   1   0  0  0  1  0  1
byte 2: 1   0   0  0  1  1  0  0
```

Packed output:

```
byte 0: 0   0   0  0  0  1  0  1
byte 1: 0   0   1  1  0  1  1  1
byte 2: 0   1   0  0  0  1  0  1
byte 3: 0   0   0  0  1  1  0  0
```

Data Dumps (Setups)

The Active Setup Data dump and Stored Setup Data dump have identical data arrangements. The table below shows the order in which the bytes are sent. The data here is shown in unpacked (8-bit) format (before it is packed into 7-bit format).

Data Byte	Data Size	Data Description
0	1-byte	Program (algorithm) ID
1,2	2-byte	Front Panel DECAY Parameter
3,4	2-byte	Front Panel DELAY Parameter
5,6	2-byte	Front Panel FX LVL Parameter
7-20	2-byte	Other microcode parameters (different for each algorithm)
21-36	1-byte	Name (16 characters)
37-40	1-byte	MIDI patch source (0-127)
41-44	1-byte	MIDI patch destination (ucode parameter 0-9)
45-48	1-byte	MIDI patch scale (-127 to +127, 2's compliment)

49 bytes total unpacked, 56 bytes packed.

The All Registers Dump format is identical, except that all 128 registers are sent in series in one SysEx message for a total length of 128 * 56=7168 packed data bytes (not counting header bytes, etc). Register 0 is sent first; register 127 last.

APPENDIX A: Parameter Map

Parameter #	APM #	Size	Description
0	1	16 bits	Front Panel DECAY Parameter
1	2	16 bits	Front Panel DELAY Parameter
2	3	16 bits	Front Panel FX LVL Parameter
3	4	16 bits	(Different for each algorithm)
4	5	16 bits	(Different for each algorithm)
5	6	16 bits	(Different for each algorithm)
6	7	16 bits	(Different for each algorithm)
7	8	16 bits	(Different for each algorithm)
8	9	16 bits	(Different for each algorithm)
9	10	16 bits	(Different for each algorithm)
10 (A hex)	–	16 bits	Effect Input Level (* used for BYPASS)
32-47 (20-2F hex)		8-bitx16	Name (16 characters)
48-51 (30-33 hex)		8-bit	MIDI patch sources 0-127
52-55 (34-37 hex)		8-bit	MIDI patch destination, ucode parameter number 0-9
56-59 (38-3B hex)		8-bit	MIDI scale factors, -127 to +127, 2's complement
60-63 (3C-3F hex)**		16-bit	MIDI patch offsets
64 (40 hex)		8-bit	Setup number
65 (41 hex)		8-bit	Algorithm number

* Changes to the "Input Level" parmeter via MIDI are not recognized by the Bypass system. Therefore, changes to this parameter may cause the Bypass function in the system to work incorrectly.
** These parameters are not stored during register save.

Note that parameters in certain algorithms which appeared in the LXP-1 have been removed from Reflex to avoid confusion. These parameters are duplicates of others and no functionality has been eliminate. In order to maintain consistency in the user interface, several duplicate parameters, were mapped to different positions in Advanced Programming Mode (APM). The following table outlines this remapping:

Algorithm	Actual Parameter	APM Parameter
A4	3	1
A5	6	1
A5	9	2
A6	8	2
A7	8	2
A8	6	1

Note that the Actual Parameter numbers listed above are "0" based as they appear throughout this document. The APM Parameter numbers listed are "1" based as they appear on the Reflex front panel. Subtract 1 from the APM Parameter numbers to get the actual parameter number.

APPENDIX B: Algorithm Parameter Definitions

This section outlines the parameters which are available for each of the system's effects algorithms. "#" is the parameter number, "Pol" refers to polarity (Bipolar or Unipolar). "Eff Num Steps" is the effective number of steps that can be used with the parameter. "Val Range" is the legal range of values that can be sent with this parameter in hex. "MRC Step Val" indicates the lowest step value output for this parameter by the MRC in hex.

Note that the MRC does not always use the maximum resolution available for a parameter. To determine the "minimum step value" (which will actually do something) subtract the high range value from the low range value and divide by the number of effective steps. For example:

If the "Val Range" = 0x8000-0xBFC0 and the "Eff Num Steps" = 8192,

0xBFC0 - 0x8000 = 0x3FC0 =16320

16320 / 8192 = 1.99 or ~2 = the minimum step value

1 Reverb Algorithm

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Mid Reverb Decay	0	Uni	16	8000-BC00	0400	RTIME
Predelay	1	Uni	8192	8000-BFC0	0040	PDLY
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Bass Multiply	3	Bi	32	4000-B800	0800	BASS
High Freq Cutoff	4	Uni	16	8000-BC00	0400	HICUT
Size	5	Uni	64	8000-BF00	0100	SIZE
Predelay Feedback	6	Bi	512	4000-BF80	0080	FDBK
Diffusion	7	Uni	256	8000-BFC0	0040	DIFF
Reflection Level	8	Uni	128	8000-BFC0	–	–
Reflection Delay	9	Uni	128	8000-BFC0	–	–

2 Plate Algorithm

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Mid Reverb Decay	0	Uni	16	8000-BC00	0400	RTIME
Predelay	1	Uni	8192	8000-BFC0	0040	PDLY
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Bass Multiply	3	Bi	32	4000-B800	0800	BASS
High Freq Cutoff	4	Uni	16	8000-BC00	0400	HICUT
Size	5	Uni	64	8000-BF00	0100	SIZE
Predelay Feedback	6	Bi	512	4000-BF80	0080	FDBK
Diffusion	7	Uni	256	8000-BFC0	0040	DIFF
Unused	8	–	–	–	–	–
Unused	9	–	–	–	–	–

3 Chorus 1 Algorithm (Flange)

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Negative Feedback	0	Uni	256	8000-BFC0	–	–
Flange Depth*	1	Uni	256	8000-BFC0	0040	DEPTH
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Right Dly Feedback	3	Bi	512	4000-BF80	0080	RFDBK
Right Delay	4	Uni	128	8000-BF80	0080	R-DLY
Shape	5	Uni	8	8000-B800	0800	SHAPE
Left Dly Feedback	6	Bi	512	4000-BF80	0080	LFDBK
Left Delay	7	Uni	128	8000-BF80	0080	L-DLY
Flange Rate	8	Uni	16	8000-BC00	0400	RATE
Unused	9	–	–	–	–	–

* A flange depth value of less than 8 samples, .25 msec, xmit val <= 0x8200 is ignored.

Delay values above 0xBD00 are limited to 32000 samples, 1 second. Resolution of 128 reflects that machine screens the 1 second delay range to 8 msec intervals.

4 Delay 2 Algorithm (Multi-Echoes)

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Unused	0	–	–	–	–	–
Group Delay	1	Uni	256	8000-BFC0	0040	GPDLY
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Feedback	3	Bi	512	4000-BF80	0080	FDBK
Left Delay	4	Uni	256	8000-BFC0	0040	L-DLY
Right Delay	5	Uni	256	8000-BFC0	0040	R-DLY
Unused	6	–	–	–	–	–
High Freq Cutoff	7	Uni	16	8000-BC00	0400	HICUT
Diffusion	8	Uni	256	8000-BFC0	0040	DIFF
Echo Rhythm	9	Uni	14	8000-B400	–	–

5 Chorus 2 Algorithm (Resonator)

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Unused	0	–	–	–	–	–
Unused	1	–	–	–	–	–
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Predelay	3	Uni	256	8000-BFC0	0040	PDLY
Low Freq Cutoff	4	Uni	256	8000-BFC0	0040	LOCUT
Shimmer	5	Uni	16	8000-BC00	0400	SHIMR
Resonance Fdbk	6	Bi	64	4000-BE00	0200	RESON
Richness	7	Uni	16	8000-BC00	0400	RICH
Slope	8	Uni	32	8000-BE00	0200	SLOPE
Tuning	9	Bi	128	4000-BF00	0100	TUNE

6 Inverse Room Algorithm

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Size	0	Uni	32	8000-BE00	0200	SIZE
Unused	1	–	–	–	–	–
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Unused	3	–	–	–	–	–
High Freq Cutoff	4	Uni	16	8000-BC00	0400	HICUT
Slope	5	Uni	32	8000-BE00	0200	SLOPE
Predelay Feedback	6	Bi	512	4000-BF80	0080	FDBK
Diffusion	7	Uni	256	8000-BFC0	0040	DIFF
Predelay	8	Uni	8192	8000-BFC0	0040	PDLY
Unused	9	–	–	–	–	–

7 Gate Algorithm

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Gate Time	0	Uni	32	8000-BE00	0200	TIME
Unused	1	–	–	–	–	–
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
Unused	3	–	–	–	–	–
High Freq Cutoff	4	Uni	16	8000-BC00	0400	HICUT
Slope	5	Uni	16	8000-BC00	0400	SLOPE
Predelay Feedback	6	Bi	512	4000-BF80	0080	FDBK
Diffusion	7	Uni	256	8000-BFC0	0040	DIFF
Predelay	8	Uni	8192	8000-BFC0	0040	PDLY
Unused	9	–	–	–	–	–

8 Delay 1 Algorithm (Chorus)

Description	#	Pol	Effective Number of Steps	Value Range	MTC Step Value	MRC Name
Unused	0	–	–	–	–	–
Delay 1	1	Uni	256	*8000-F700	0040	DELAY
Effects Level	2	Uni	256	8000-BFC0	0040	FXLVL
High Freq Cutoff	3	Uni	16	8000-BC00	0400	HICUT
Delay 2 Spread	4	Uni	128	8000-BF80	0080	DLY-2
Delay 3 Spread	5	Uni	128	8000-BF80	0080	DLY-3
Feedback	6	Bi	512	4000-BF80	0080	FDBK3
Diffusion	7	Uni	256	8000-BFC0	0040	DIFF
Chorus Rate	8	Uni	16	8000-BC00	0400	RATE
Echo Rhythm	9	Uni	14	8000-B400	–	–

* The total delay available in Reflex is 1612ms with each tap set at 20.3ms intervals, as opposed to the 1000ms available in the LXP-1. Note that the MRC only produces values up to the 0xBFC0 value supported by the LXP-1.

Note that the parameter accessed by the MRC: FDBK3 fader is called FDBK1 in the Reflex User Guide.

MIDI Implementation Chart

Lexicon Reflex
Digital Effects System

Function		Transmitted	Recognized	Remarks
Basic Channel	Default Changed	1 1-16	1 1-16	memorized can be set from APM
Mode	Default Messages Altered		X 3 X	
Note Number		X	0-127	used as controller
Velocity	Note ON Note OFF	X	O 9n v = 0-127	used as controller
After Touch	Keys Channel	X X	X O	
Pitch Bend		X	O	
Control Change	1-119	OX	OX	
Program Change	True #	X	0-127	
System Exclusive	Lexicon Real-time non Real-time	O X X	O X X	
System Common	:Song Pos :Song Sel :Tune	X X X	X X X	
System Real Time	:Clock :Commands	X X	O X	used as controller
Aux Messages	:Local ON/OFF :All Notes OFF :Active Sense :Reset :Reset All Controllers	X X X X X	X X X O X	
Notes				

Mode 1: OMNI ON, POLY Mode 2: OMNI ON, MONO **O : Yes** **OX: Selectable**
Mode 3: OMNI OFF, POLY Mode 4: OMNI OFF, MONO **X : No**

Lexicon, Inc.
3 Oak Park
Bedford MA 01730-1441
Tel: 617 280-0300
Fax: 617 280-0490

Lexicon Part No. 070-10748 Rev 1

Printed in U.S.A.