Lîla

A Multi-Channel Instrument for Spatialization of Loops, Delays, and Sound Files

Shahrokh Yadegari and Mammad Zadeh

December 30, 2025

NOTE: This is a very incomplete and an in-progress document and has been updated for Lila 0.72.

The html version of this document is available by clicking here

Contents

1 Introduction

Lîla is a computer music instrument currently implemented in Pure Data version 0-55-2 and above. The word "*lîla*" is an old Sanskrit word signifying divine play, the play of destruction and creation, or the play of presence in the moment. The computer music instrument *Lîla* is built based on simple analog processes (e.g., loop, delay, ring modulation, and feedback) whose parameters are controlled precisely by a performative action or through messages. *Lîla* samples and transforms the acoustic material played in real-time based on the actions of the *Lîla* player; the acoustic performer can improvise more material on this newly created sound. This becomes a continual and circular process, and through the use of delays and feedback, the resuting sound can become complex quickly. The precise real-time control of the parameters allows the *Lîla* improvisor to participate in both micro and macro level of musical formations. Thus, the computer not only can act as agent of form in macro structure of time (such as it is in music involving tape music) and lead the acoustic performer, but also provides a musical context in which a human improvisor, using the computer as an instrument, can accompany and respond to the acoustic material. Thus, the acoustic performer can have the same form of musical freedom which he or she enjoys in a traditional setting in an augmented expressive language.

Network extensions have been added to *Lîla* so that its performer could control multiple instances of the program over the network, while *Lîla* compensates for the actions of the performer based on the intrinsic network delays. I am interested in exploration of the play with space over the network, in the same way that I am able to play with delay in a single location, to turn the physical distance into an ephemeral yet malleable artistic parameter.

1.1 Installation

Currently *Lîla* is only available for Mac OS X. One can download the application or clone the git repository.

1.2 Download Application

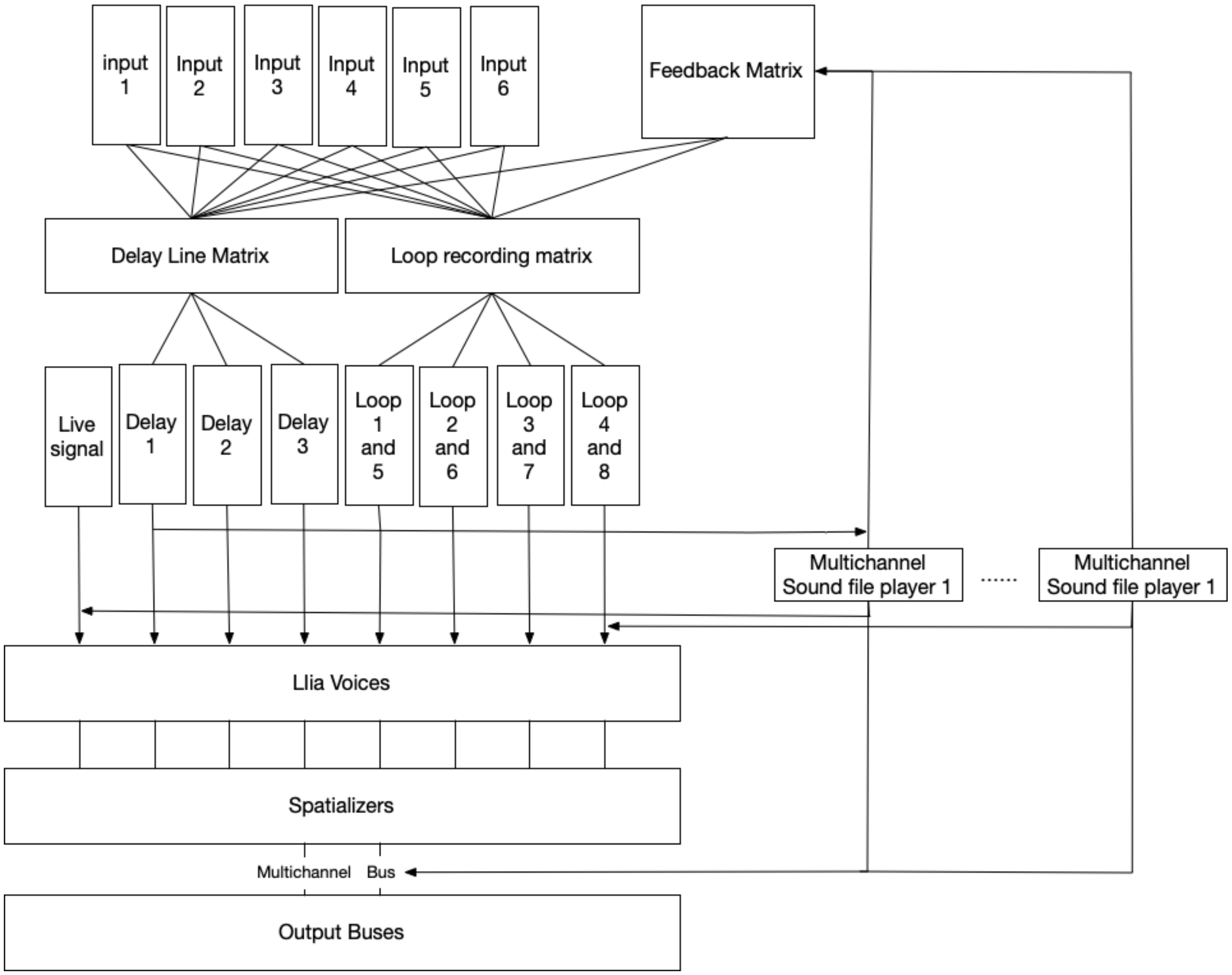
Download current version of *Lîla* for Mac OS App
Click here for older versions

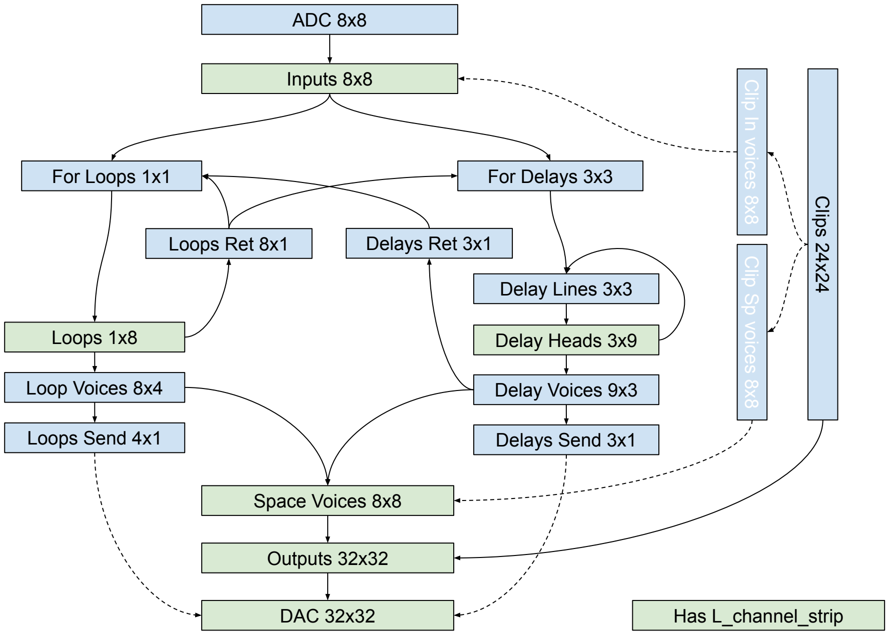
1.2.1 Via Git

```
git clone --recursive git@gitlab.com:Yadegari/Lila.git

cd Lila

git submodule update --init --recursive
```

2 Theory of Operation

TBD


input 1
Input 2
Input 3
Input 4
Input 5
Input 6
Feedback Matrix
Delay Line Matrix
Loop recording matrix
Live signal
Delay 1
Delay 2
Delay 3
Loop 1 and 5
Loop 2 and 6
Loop 3 and 7
Loop 4 and 8
Multichannel Sound file player 1
......
Multichannel Sound file player 1
Llia Voices
Spatializers
Multichannel Bus
Output Buses



3 Performing with *Lîla*

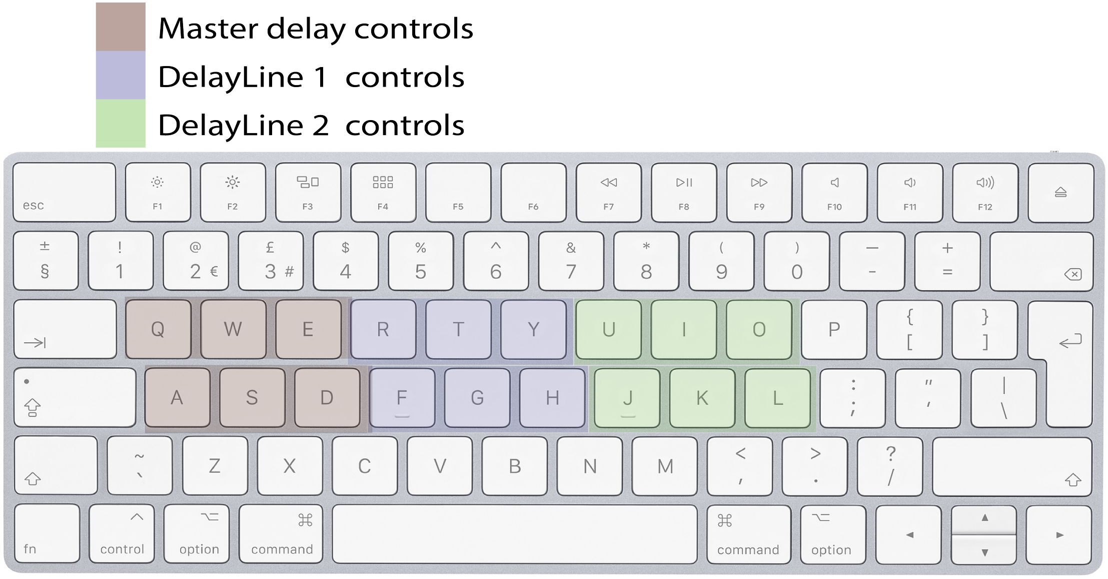
In this section the performative interface of Lila is discussed.

3.1 Performing with Delays

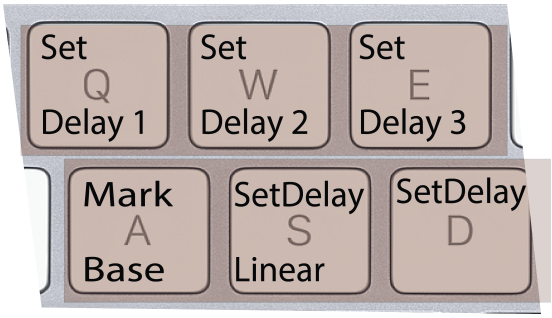
Lîla provides 3 separate delay lines each with 3 read heads (instances). The value of each delay can be controlled with either performative actions or through message passing. By default inputs 1 and 2 are sent to delay line 1; inputs 3 and 4 are sent to delay line 2, and inputs 5 and 6 are sent to delay line 3.

The concept of keyboard control of delays is that a key is pushed to **Mark** the base of a delay line, Then a **Set** key is pushed to set the duration of the delay, which will be the time difference between the *Mark*ing of the delay line and the *Set*ing of it. There are various performatively useful ways to *mark* and *set* the multitude of delay lines.

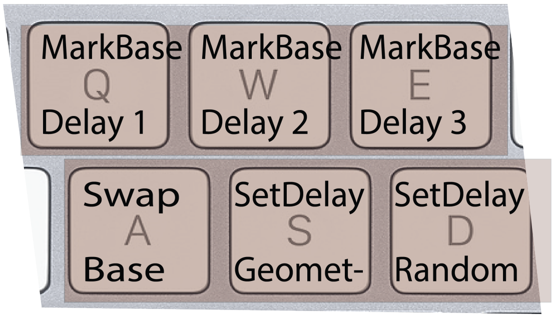
Six keys are used for each set of controls. One set (keys Q, W, E, A, S, D) is used master set to control all the instances of all the delay lines. One set (Keys R, T, Y, F, G, H) are used to control instances of delay line 1, and another set (keys U, I, O, J, K, L) is used to control the instances of delay line 2. The delay line 3 is only controlled by the master control (because of lack of space on the keyboard), but it is possible to set its values separately through scripting.



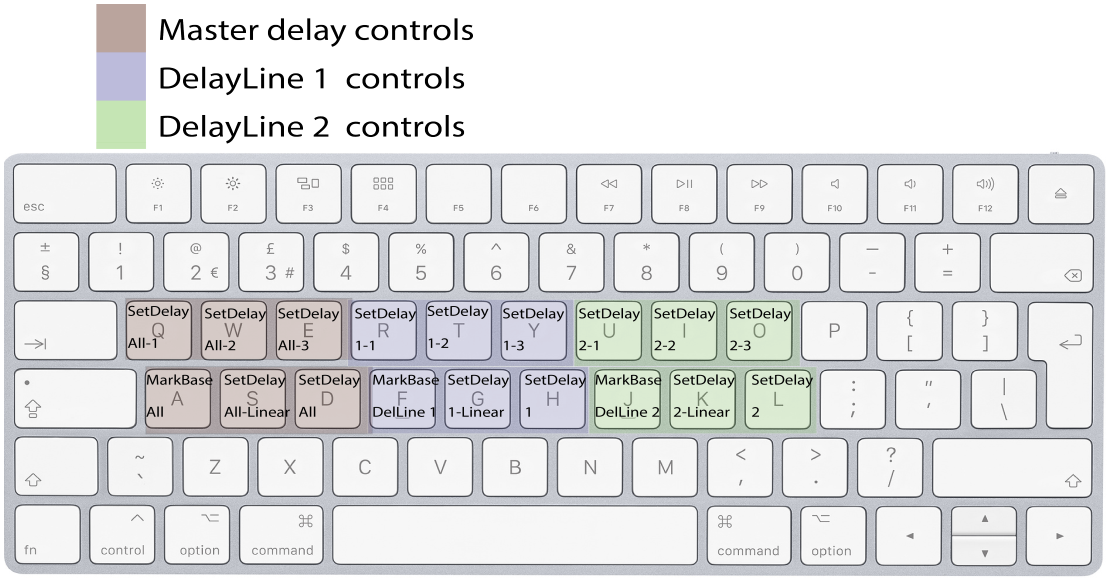
Most users will only need to use the master controls. The key ”a” is used to mark the base of all the delay lines. The key ’q’ will set the duration of read head 1 for all the delay lines. The key ’w’ will set the duration of read head 2 for all the delay lines, and the key ’e’ will set the duration of read head 3 for all the delay lines. The key ’s’ will set the value of the delay lines linearly, with the first read head duration being set according to the moment the key ’s’ was pushed, and the delay line duration for read head 2 and 3 will be twice and three times the duration of the first delay read head respectively. The key ’d’ will set the duration of all the delay lines corresponding to the base of each delay read head.



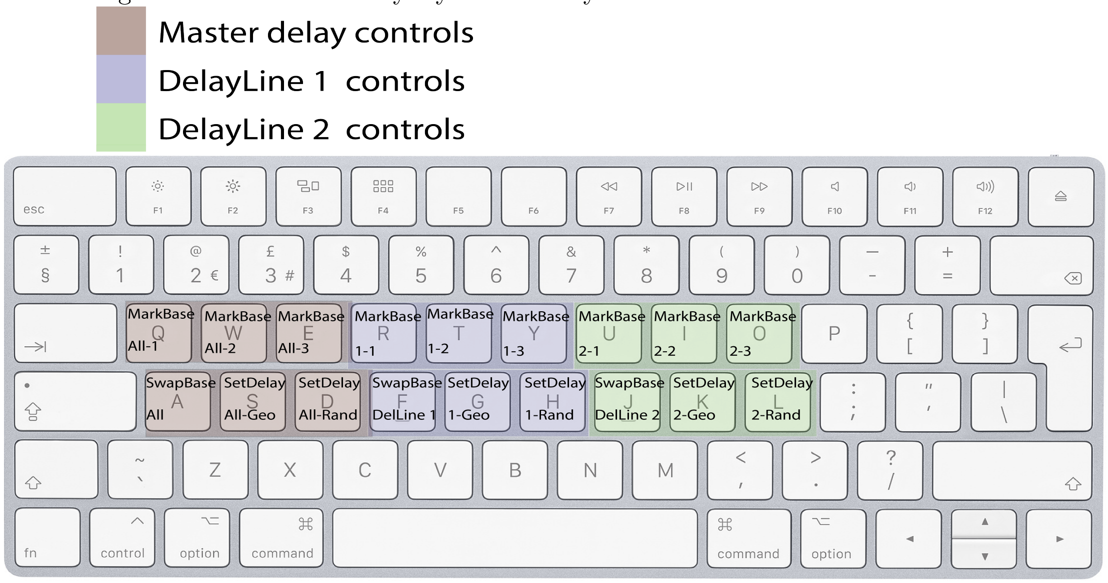
Using shift keys, you can set the base for each instance separately. Letter 'Q' (shift q) sets the base of first instances of all delay lines. 'W' sets the base for the 2nd instances of all delay lines, and 'E' sets the base of the 3rd instances of all delay lines. Lila has a single memory for the delay base locations. 'A', swaps the base of all instances of all delay lines with previously set values Letter 'S' (shift 's') arranges the delays geometrically where delay duration of read head 2 is twice the duration of read head 1, and duration of read head 3 is 4 times the duration of read head 1.



As mentioned above one can set the base and duration of the instances of each delay line separately as well in case the performer chooses to have different delay values for different inputs. Below diagram shows the non-shifted key layout for delay control:



Below diagram shows the shift key layout for delay control:



3.2 Performing with Loops

4 Scripting for *Lîla*

In this section the scripting interface of *Lîla* is discussed. All values in *Lîla* could be controlled by messages. This model is a fundamental element in automation of *Lîla* , as well as for it network communication. Message names for various objects are defined as follows:

```
L_<module>_<inst>_<param>[_attr]
```

where ¡module¿ is the module name, ¡inst¿ is the channel or instance number,

4.1 Scripting of Delays

4.2 Scripting of Loops

```
L_loop_#_<cmd>[_attr]
```

```
L_loop_#_<cmd>[_attr]
```

4.3 Scripting of Sound Files

5 *Lîla* References

Lîla operates on a message passing model. Almost all affordances of *Lîla* can be controlled by sending messages to various subsystems, and it send various messages about its operations as well. *Lîla* also has various accessable variable. All names follow the following conventions: [1]

```
L_<module>[s]_[#_]_<cmd>
```

- where `<module>` is the name of *Lîla* module
- the added 's' after a module name implies that the opertation should be applied to all the channels of that module
- if a module has multiple channels the [_#] refers to the specific channel of that module
- ¡cmd¿ will be the desired operation

as examples the command below will set the first input volume to 100

```
L_input_1_vol 100
```

and this command will set all the input values to 0

```
L_inputs_vol 0
```

in many cases a duration in milliseconds can be passed for the operation. As an example, the following will fade the master volume to 0 in 4 seconds:

```
L_master_vol 0 4000
```

[1] [ ] notation denotes an optional item

5.1 Channel Strip naming convention

If a module has a channel strip the following commands and naming convention are available:

Receiving Methods and Signals

Name	Type	Args	Notes	Example
L_<module>_#_sat_active	bool	0 or 1	turn saturation on/off	L_input_1_sat_active 0
L_<module>_#_sat_reset	bang	n/a	reset saturation	
L_<module>_#_sat_gain	float	val [dur]	set percentage 0 to 100	L_input_1_sat_gain 80 1000
L_<module>_#_sat_mix	float	val [dur]	mix level	L_input_1_sat_mix 80 1000
L_<module>_#_sat_shape	float	val [dur]	shape of saturation	L_input_1_sat_shape 50 1000
L_<module>_#_mod_active	bool	0 or 1	turn modulation on/off	L_input_1_mod_active 0
L_<module>_#_mod_reset	bang	n/a	reset modulation	
L_<module>_#_mod_mix	float	val [dur]	mix level	L_input_1_mod_mix 80 1000
L_<module>_#_sat_freq	float	val [dur]	freq of modulation	L_input_1_mod_freq 7 1000
L_<module>_#_mod_sync	bool	0 or 1	MZ	i L_input_1_mod_sync 1
L_<module>_#_delay_active	bool	0 or 1	turn delay on/off	L_input_1_delay_active 0
L_<module>_#_delay_reset	bang	n/a	reset delay	
L_<module>_#_delay	float	flt [dur]	delay len	L_input_1_delay 10
L_<module>_#_delay_mix	float	val [dur]	mix level	L_input_1_delay_mix 80 1000
L_<module>_#_delay_lfo	int	0-3	MZ	L_input_1_delay_lfo 3

5.2 Inputs

Receiving Methods and Signals

Name	Type	Args	Notes	Example
L_inputs_vol	float	db [dur]	set volume of all inputs	L_inputs_vol 100 L_inputs_vol 0 1000
L_inputs_ch	int	n/a	set dac number for all input channels	L_inputs_ch 3 L_inputs_ch 0 (all off)
L_input_#_vol	float	db [dur]	set volume of a specific channel	L_input_2_vol 100 2000
L_input_#_ch	int	n/a	set dac number of a specific input channel	L_input_2_ch 10

Value Sends and Signals

L_input_#_input_sig	signal		add signal to a specific input channel	throw∼ L_input_#_input_sig

5.3 Outputs

Receiving Methods and Signals

Name	Type	Args	Notes	Example
L_outputs_vol	float	db [dur]	set volume of all outputs	L_outputs_vol 100 L_outputs_vol 0 1000
L_outputs_ch	int	n/a	set dac number for all output channels	L_outputs_ch 3 L_outputs_ch 0 (all off)
L_output_#_vol	float	db [dur]	set volume of a specific channel	L_output_2_vol 100 2000
L_output_#_ch	int	n/a	set dac number of a specific output channel	L_output_2_ch 10

Value Sends and Signals

L_output_#_output_sig	signal		add signal to a specific output channel	throw∼ L_output_#_output_sig

5.4 Voices

Receiving Methods and Signals

Name	Type	Args	Notes	Example
L_voices_vol	float	db [dur]	set volume of all voices	L_voices_vol 100 L_voices_vol 0 1000
L_voice_#_vol	float	db [dur]	set volume of a specific channel	L_voice_2_vol 100 2000

Value Sends and Signals

L_voice_#_input_sig	signal		add signal to a specific voice channel	throw∼ L_voice_#_input_sig

5.5 Loops

Receiving Methods and Signals

Name	Type	Args	Notes	Example
L_loop_#_browse	bang	n/a	load file into loop through panel	
L_loop_#_dbx	float	beats	for SetDelays: length in beats	L_loop_1_dbx 4
L_loop_#_dbx_wait	float	beats	for SetDelays: wait in beats before setting delay	L_loop_1_dbx_wait 16
L_loop_#_dbx_dest	int	1-3	for SetDelays: delay destination	L_loop_1_dbx_dest 1
L_loop_#_file	string	filename	set filename for read/write	L_loop_3_file loops/loop_3.wav L_loop_4_file layer4.wav
L_loop_#_fullwrite	str; bang	filename; n/a	write loop to given filename; a bang writes to preset filename	L_loop_1_fullwrite path/loop_one.wav
L_loop_#_init	bang	n/a	initialize loop settings	
L_loop_#_label	string	name	set loop channel label	L_loop_2_label HiHat
L_loop_#_len	float	ms [dur]	set loop length (ms)	L_loop_1_len 4000 L_loop_1_len 2000 1000
L_loop_#_len_beat	float	Blen [dur]	set loop length (beats per loop)	L_loop_1_len_beat 8 L_loop_1_len_beat 16 1000
L_loop_#_len_samp	float	Slen [dur]	set loop length (samples)	L_loop_1_len_samp 44100 L_loop_1_len_samp 88200 1000
L_loop_#_len_set	float	ms [dur]	set loop length (ms)	L_loop_1_len_set 4000 L_loop_1_len_set 2000 1000
L_loop_#_mute	bool	0 or 1	turn loop mute on/off	L_loop_1_mute 1
L_loop_#_mute_reset	bang	n/a	reset mute to off	
L_loop_#_offset	float	ms	offset starting point of loop playback	L_loop_2_offset 50
L_loop_#_ol	float	ms	set loop overlap	L_loop_3_ol 20
L_loop_#_ol_reset	bang	n/a	reset overlap to default	
L_loop_#_phase	float	val	set phase of loop	L_loop_4_phase 1000
L_loop_#_phones_pan	val [dur]	-100(L) to 100(R)	pan loop in monitor	L_loop_5_phones_pan 50 L_loop_1_phones_pan -100 2000

L_loop_#_phones_pan_reset	bang	n/a	reset loop panning in monitor to default	
L_loop_#_phones_solo	bool	0 or 1	turn solo on/off in monitor	L_loop_6_phones_solo 1
L_loop_#_phones_solo_reset	bang	n/a	reset solo to off in monitor	1
L_loop_#_phones_vol	float	dB [dur]	set loop volume in monitor	L_loop_7_phones_vol 100 L_loop_8_phones_vol 0 1000
L_loop_#_phones_vol_reset	bang	n/a	reset loop volume in monitor	L_loop_1_phones_vol_reset bang
L_loop_#_play	bool	0 or 1	start/stop playback of loop	L_loop_1_play 1
L_loop_#_play_sw	bang	n/a	toggle playback state (switch)	L_loop_1_play_sw bang
L_loop_#_play	bang	n/a	stop playback of loop	L_loop_1_stop bang
L_loop_#_qbx	int	beats	set recording length in beats (0 = indefinite)	L_loop_2_qbx 8
L_loop_#_qbx_reset	bang	n/a	reset number of beats to record	L_loop_2_qbx_reset bang
L_loop_#_read	bang	n/a	load preset file	L_loop_2_read bang
L_loop_#_read_file	string	filename	load given file	L_loop_3_read_file snare.wav
L_loop_#_rec	bool	0 or 1	start/stop recording of loop	L_loop_3_rec 1
L_loop_#_rec_start_on_bar	bool	0 or 1	1 = start recording on bar division	L_loop_4_rec_start_on_bar 1
L_loop_#_rec_start_on_bar_reset	bang	n/a	reset Rec-On-Bar to off	L_loop_5_rec_start_on_bar_re bang
L_loop_#_rec_sw	bang	n/a	toggle recording state (switch)	L_loop_6_rec_sw bang
L_loop_#_reset	bang	n/a	reset loop to default state (clears file)	L_loop_2_reset bang
L_loop_#_ret_dest	int	1-3	set return/destination (delay line) for loop output	L_loop_3_ret_dest 2
L_loop_#_ret_dest_reset	bang	n/a	reset loop return destination to default	L_loop_1_ret_dest_reset bang
L_loop_#_ret_dest_set	int	1-3	set return/destination (delay line) for loop output	L_loop_2_ret_dest_set 1
L_loop_#_reverse	bool	0 or 1	toggle reverse playback of loop	L_loop_4_reverse 1

L_loop_#_reverse_reset	bang	n/a	reset reverse playback to normal (off)	L_loop_1_reverse_reset bang
L_loop_#_trans	float	val	set transposition amount (semitones, +/-)	L_loop_4_trans 12
L_loop_#_trans_mix	float	val(0-100) [dur]	mix level for transposed loop	L_loop_5_trans_mix 50 L_loop_6_trans_mix 0 2000
L_loop_#_trans_mix_reset	bang	n/a	reset transposed loop mix level to 0	L_loop_5_trans_mix_reset bang
L_loop_#_trans_phones_pan	val [dur]	-100(L) to 100(R)	pan transposed loop in monitor	L_loop_7_trans_phones_pan 50 L_loop_8_trans_phones_pan -100 5000
L_loop_#_trans_phones_pan_reset	bang	n/a	reset transposed loop panning in monitor to 0	L_loop_8_trans_phones_pan_ bang
L_loop_#_trans_phones_solo	bool	0 or 1	turn solo on/off in monitor for transposed loop	L_loop_1_trans_phones_solo 1
L_loop_#_trans_phones_solo_reset	bang	n/a	reset transposed loop solo to off in monitor	L_loop_1_trans_phones_solo_ bang
L_loop_#_trans_phones_vol	float	dB [dur]	set volume for transposed loop in monitor	L_loop_2_trans_phones_vol 100 L_loop_3_trans_phones_vol 0 1000
L_loop_#_trans_phones_vol_reset	bang	n/a	reset transposed loop monitor volume to 0	L_loop_3_trans_phones_vol_r bang
L_loop_#_trans_semi	float	semitones	transpose loop by specified semitones	L_loop_4_trans_semi -2 L_loop_5_trans_semi 12
L_loop_#_trans_semi_reset	bang	n/a	reset semitone transpose to 0	L_loop_4_trans_semi_reset bang
L_loop_#_trans_spd	float	ratio or %	set playback speed for transposed loop (1 = normal)	L_loop_5_trans_spd 0.5
L_loop_#_vol	float	dB [dur]	set loop volume	L_loop_6_vol 100 L_loop_7_vol 0 1000
L_loop_#_write	bang	n/a	write current loop to a specified file	L_loop_8_write bang
L_loops_browse	bang	n/a	open eight directory windows (in sequence) to manually load files into loops 1-8	L_loops_browse bang

Value Sends and Signals

L_loop_#_recorded	bool	0 or 1	0 = empty, 1 = recorded	r L_loop_1_recorded_do 1
L_loop_#_recorded_val	bool	0 or 1	0 = empty, 1 = recorded	r L_loop_1_recorded_val 1
L_loop_1_sig				