Dsp.rack: Laptop-based Modular, Programmable Digital Signal Processing and Mixing for Live Performance

William Kleinsasser
Towson University
Department of Music
8000 York Road
Baltimore, MD 21252 USA
wkleinsasser@towson.edu

ABSTRACT

This document describes modular software supporting live signal processing and sound file playback within the Max/MSP environment. Dsp.rack integrates signal processing, memory buffer recording, and pre-recorded multi-channel file playback using an interconnected, programmable signal flow matrix, and an eight-channel i/o format.

KEYWORDS

Digital signal processing, Max/MSP, computer music performance, matrix routing, live performance processing.

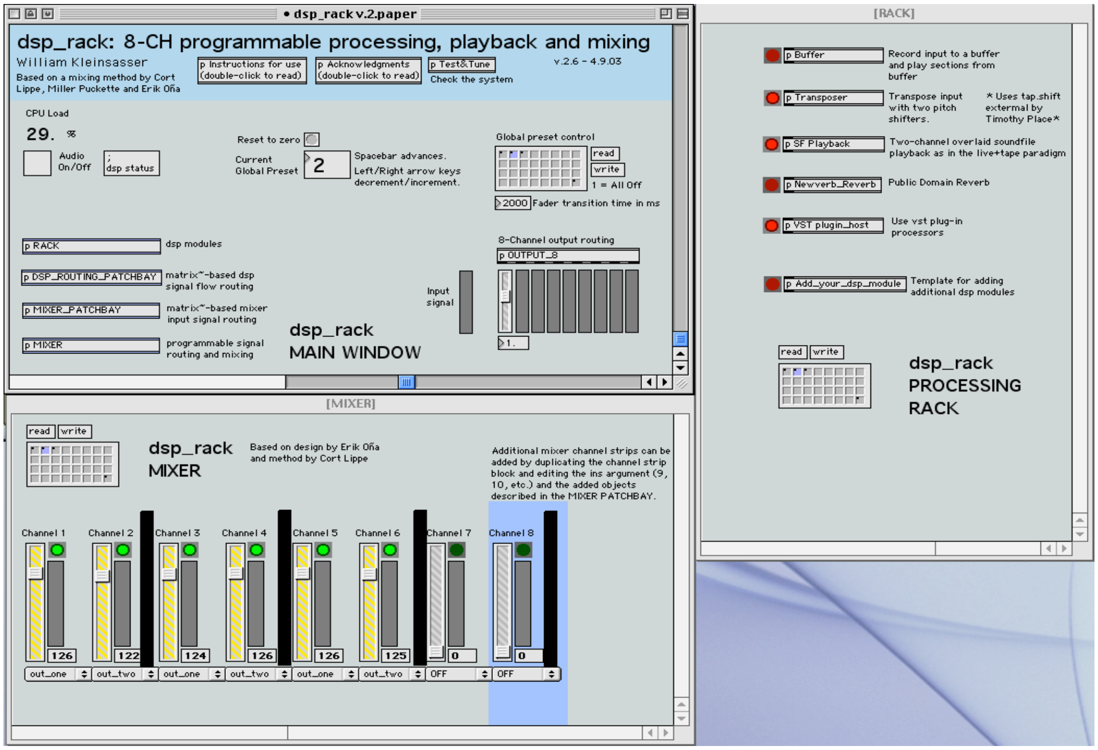
1. INTRODUCTION

Dsp.rack is a suite of Max/MSP modules that run on a Macintosh Powerbook, iBook, or desktop computer with a G3 500 mHz or faster CPU. Dsp.rack uses the familiar paradigm of combined mixer, patch bay, and signal processors for integrating electronic music with live performance. Dsp.rack was developed to take advantage of the familiarity of this paradigm and the decades of performance practice related to it. Building on the flexibility offered by software-based systems, Dsp.rack integrates the functions of programmable mixing, routing, and audio processing along with the ability to play overlaid, pre-recorded sound files. Dsp.rack was designed to offer a familiar, flexible, and open-ended entry point to composers, performers, students, and teachers.

2. THE DESIGN

Dsp.rack is available in two versions which offer beginning and more advanced environments for live performance. Dsp.rack version 1 uses a menu-driven crossbar method for routing signals. This version offers a flexible and simple approach to integrating signal input, routing, processing, mixing, and output. Version 2 uses the matrix~ object for routing that supports programmable, complex, signal flow combinations. Having been developed in Max/MSP, Dsp.rack also benefits from the open sharing of resources that comes with that environment.

A basic set of processing modules is included with the distribution of Dsp.rack and a mini-tutorial on integrating


• dsp_rack v.2.paper

dsp_rack: 8-CH programmable processing, playback and mixing
William Kleinsasser
Based on a mixing method by Cort Lippe, Miller Puckette and Erik Oña
p Instructions for use (double-click to read)
p Acknowledgments (double-click to read)
p Test&Tune
Check the system
v.2.6 - 4.9.03

CPU Load
29. %
Audio On/Off
; dsp status

Reset to zero
Current Global Preset 2
Spacebar advances.
Left/Right arrow keys decrement/increment.

Global preset control
read
write
1 = All Off
2000 Fader transition time in ms

p RACK — dsp modules
p DSP_ROUTING_PATCHBAY — matrix~-based dsp signal flow routing
p MIXER_PATCHBAY — matrix~-based mixer input signal routing
p MIXER — programmable signal routing and mixing

dsp_rack
MAIN WINDOW

Input signal
8-Channel output routing
p OUTPUT_8
1.

[MIXER]

read
write
dsp_rack
MIXER
Based on design by Erik Oña and method by Cort Lippe

Additional mixer channel strips can be added by duplicating the channel strip block and editing the ins argument (9, 10, etc.) and the added objects described in the MIXER PATCHBAY.

Channel 1	Channel 2	Channel 3	Channel 4	Channel 5	Channel 6	Channel 7	Channel 8
126	122	124	126	126	125	0	0
out_one	out_two	out_one	out_two	out_one	out_two	OFF	OFF

[RACK]

p Buffer — Record input to a buffer and play sections from buffer
p Transposer — Transpose input with two pitch shifters. * Uses tap.shift external by Timothy Place*
p SF Playback — Two-channel overlaid soundfile playback as in the live+tape paradigm
p Newverb_Reverb — Public Domain Reverb
p VST plugin_host — Use vst plug-in processors
p Add_your_dsp_module — Template for adding additional dsp modules

read
write
dsp_rack
PROCESSING
RACK

Figure 1: dsp.rack screens

additional user-designed modules is provided. The mixer and patch bay are extendable and limited only by screen saturation and processing speed of the computer.

Running on a Powerbook with an eight-channel i/o converter like the RME Hammerfall or MUTO 828, Dsp.rack can support eight independent input and output channels for processing. With other i/o hardware, like the MOTU 2408, it can support up to 24 channels. This makes Dsp.rack capable of instrumental and vocal ensemble processing with multi-channel output.

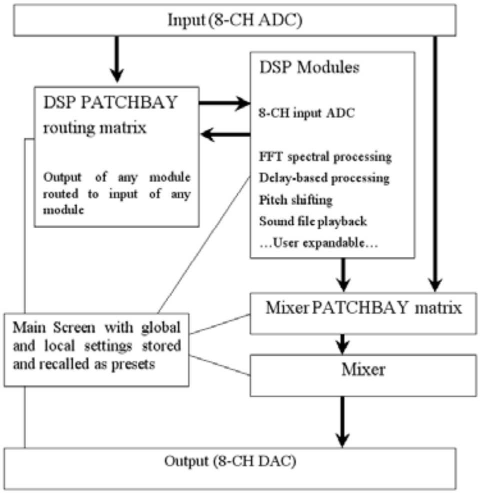


Figure 2: dsp.rack block diagram

3. INTEGRATED SIGNAL PROCESSING AND PRE-RECORDED SOUND FILE PLAYBACK

The integrated performer+tape paradigm that flourished after 1960 offers a model of musical expression that expands the capabilities of acoustic music through integration with electronic studio environment. Composers have produced a repertoire that presents acoustic performance in the context of technologically transformed music on tape but the synchronization issues involved in performer+tape music remain a concern in these works. Dsp.rack is designed to support live interactive signal processing as well as performer+tape repertoire.

This is done by offering the ability to present pre-recorded, overlapping sound files using a method for mixed overlaying that enables performance timing flexibility. The sound file player module loads and plays sound files using four independent multi-channel players. Sound files can either be routed directly out to the sound system or, using the flexible signal flow matrix, they can be routed to the inputs of the other processing modules. Dsp.rack can layer sound files with as many channels as the i/o supports depending on sufficient drive speed, i/o buffering, and CPU loading.

4. PERFORMANCE AND CPU LOAD

CPU load is directly related to the processing intensity and number of simultaneous modules used as well as the i/o vector sizes. Running several simultaneous dsp modules, an 8-channel mixer, and 8-channel i/o, Dsp.rack uses about 35% of the CPU on a 1G G4 Powerbook. The same setup uses about 75% of a 500 mHz G3 Powerbook. Dsp.rack uses the mute object for enabling and disabling each individual dsp processor which is useful for handling collections of processor-intensive modules.

Dsp.rack provides a path to familiar, personally expandable tools for integrating computer music with live performance and it is hoped that it will prove attractive to composers, performers, students, and those who teach others entering the field of live electro-acoustic music.

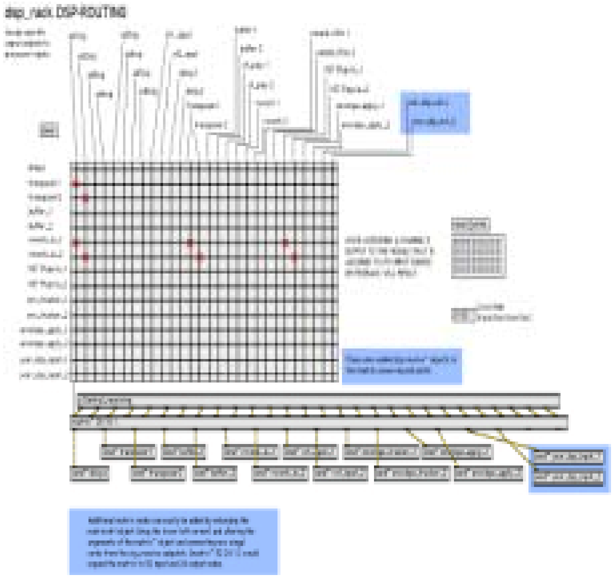


Figure 3: Version 2 matrix-driven routing

5. ACKNOWLEDGMENTS

The audio processing in Dsp.rack is based on standard-issue Max/MSP objects with the exception of the tap.shift pitch shifting object which is distributed with Dsp.rack by permission from its programmer, Timothy Place.

Dsp.rack owes to the following Max/MSP developers who have offered models and suggestions during development: Cort Lippe, Miller Puckette, and Erik Ona who developed models for crossbar mixing and routing methods using menu-driven send/receive signal flow. The approach of modular dsp functions in an integrated software environment relates to work by Cort Lippe (*compositions*) and Zack Settel (*multi effects processor, Jimmies*) Christopher Dobrian and Cort Lippe offered help on the buffer writing method and other audio handling. Daniel Koppelman provided the preset advancing method. The sound file playback and delay methods were developed in order to help, and deriving help from, my students Brian Comotto, Daniel Hope, Ljiljana Jovanovic, Scott Leake, and Nicholas Schoeb. Thanks to Miller Puckette and David Zicarelli for developing Max and Max/MSP and to the Max/MSP developers who share their solutions and ideas.

6. REFERENCES

[1] Dobrian, C. Programming New real-time DSP Possibilities with MSP, *Proceedings of the 2nd COST G-6 Workshop on Digital Audio Effects (DAFx99)*, NTNU, Trondheim, December, 1999

[2] Lippe, C. A Look at Performer/Machine Interaction Using Real-time Systems, *Proceedings of the International Computer Music Conference*, Hong Kong, 1996

[3] Lippe, C. A Composition for Clarinet and real-time Signal Processing: Using Max on the IRCAM Signal Processing Workstation, *Proceedings of the 10th Italian Colloquium on Computer Music*, Milan, 1993

[4] Lippe, C. Music for Piano and Computer: A Description, *Information Processing Society of Japan SIG Notes*, Volume 97, Number 122, 1997

[5] Place, T. tap.tools, Silicon Prairie Intermedia, http://www.sp-intermedia.com/downloads/index.html

[6] Puckette, M. New Public-Domain Realizations of Standard Pieces for Instruments and Live Electronics, *Proceedings, International Computer MusicConference*, 2001

[7] Rowe, R. *Interactive Music Systems: Machine Listening and Composing*, The MIT Press, Cambridge, 1993

[8] Settel, Zack, Jimmies, lecture/demonstration at the Max/MSP workshop at the University at Buffalo, June, 1999, Buffalo, NY

[9] Zicarelli, D. An Extensible Real-Time Signal Processing Environment for Max, *Proceedings of the International Computer Music Conference*, Ann Arbor, 1998[i]

[i] More about Dsp.rack can be found at: http://concert.towson.edu/WK/dsp.rack