MACIAS: A Comprehensive Performance System that Intelligently Couples Enhanced Stringed Instruments

Keith A. McMillen
keith@beamfoundation.org
December 20, 2006

Abstract

MACIAS is an advanced environment for the performance of live electronic music that integrates multiple stringed instruments, synthesis, processing, audio capture, notation, and timbral management with a scaleable control structure. The software provides a foundation of necessary audio, event and control elements that support multiple musical styles across a variety of idioms. An extensive modulation matrix combined with conditional execution menus allows detailed control of thousands of targets from hundreds of sources. Symmetric interface windows create a knowable system that encourages experimentation and refinement with a wide range of specificity while retaining a high degree of subtlety. A hierarchical nameable preset/library storage system responds to triggers from the composition or performance to allow rapid transitions or incremental evolution to a sound or process. Finally a graphical representation system presents the performers with a mutable notation system as well as other visual indicators.

How to create live electronic music with a traditional ensemble?

While seemingly a straightforward problem, creating a well tuned, versatile, comprehensive, and stage worthy system required significant invention and iteration. Four distinct versions of the MACIAS (Multiple Access Computerized Improvisationally Aligned Sequencer) performance system existed prior to the current system. As with all projects, the scope and level of performance grew with each generation. The intricacies of each component and the desired level of integration took over 25 years to identify, refine and implement. In summary the goals of the intended system included:

- A clear and expressive range of instrument audio
- Support of multiple musicians
- A high level of integration and interaction
- Hierarchical and thorough control and recall
- Gesture, pitch, and beat tracking
- Synthesis
- Processing
- Large flat modulation matrices

Violin System 1981

- Flexible live audio capture
- Display and score representation
- Comprehensive yet extendable
- Satisfying and endearing

At the start of this project (February 1979) programmable audio devices were not readily available. This prompted my design of the MPX 820 (the first programmable MIDI Mixer released by AKAI), a recallable patch bay (MB 76), and instrument effects systems such as the Peavey Cyber-System. The first version of MACIAS used digitally controlled analog synthesis and audio processing. This early version ran on an S100 bus 8080 rack mount computer.

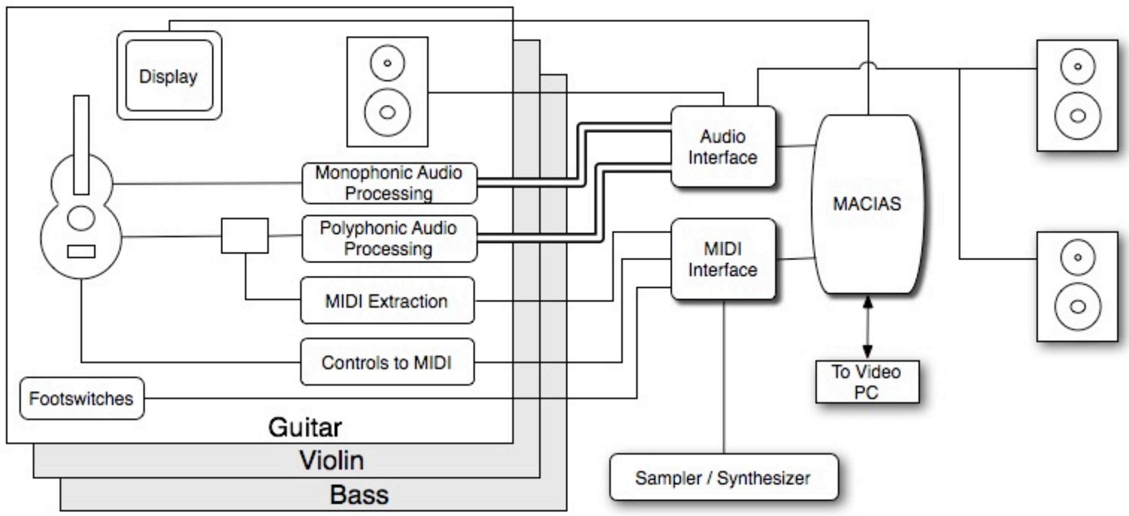
David Wessel introduced me to Opcode's MAX in 1988 and all subsequent versions of MACIAS have grown from patches crafted in Max, and later Max/MSP. Over the years the software has grown to over 250 UI windows and 480 subpatches and abstractions. However, the system still remains lightweight enough to reliably perform on today's hardware. While earlier versions of the system had to divide event, GUI and MSP audio processing on different CPUs connected via Ethernet, presently the entire application runs on a single PowerPC Macintosh. Executable program size (with no data or audio buffers) is 125Mb.

Currently, external hardware is heavily used, and accounts for roughly 3500 MIPs of task specific processing. Two racks contain the external processors and interfaces. Keeping some of these functions in external hardware frees the burden on precious host cycles. However, as PCs grow faster further integration will be explored. [1]

Individual Instrument Resources

There is a strong desire to retain the qualities of traditional stringed instruments (and the virtuosity of their human players) while extending the instruments' functions and timbre.[2] The bowed and plucked string instruments were the focus of a dozen years of R&D at Zeta Music.[3] The result is a family of highly playable instruments boasting a clear and expressive range with polyphonic audio outputs. These outputs enable extensive signal processing per string, including pitch and amplitude tracking, due to excellent cross-talk isolation between the strings and significant bow and body noise rejection.

Each of the three instruments (violin, upright bass, and guitar as played by TrioMetrik) has a similar audio processing chain consisting of two signal paths, one polyphonic and one monophonic. The polyphonic output of the enhanced instruments drives a Zeta Synthony MIDI extractor and a Roland V series polyphonic signal processor. Each instrument also produces a monophonic signal from either discrete or summed pickups. This mono signal is processed by a separate set of outboard devices. The resulting set of 2 stereo paths per instrument enters the CPU via a Firewire interface. MIDI from the instruments, footswitches and other controllers enters the CPU via a USB interface.



Each instruments is capable of generating some or all of the following control information:

- Pitch
- Dynamics
- String bend
- Pick position/Timbre index
- Joystick
- XYZ accelerometer
- Pressure sensors
- Various switches and potentiometers

Additional information is derived from the bows used by the violinist and bassist. Bow tension, grip pressure, pots, switches and XYZ accelerometers are sent via a wireless network to the host CPU. Ethernet connects the MACIAS system to video and other performance based functions.

Modulation Matrix

Once the audio and control signals enter the system it is available to the hundreds of processors and modulators. Three separate bussing matrices exist; audio, control, and trigger. The control and trigger matrices currently have hundreds of sources and well over a thousand selectable destinations.[4]

As resources and processes were added to the system the need for a flat extensible modulation space became clear. External sources include the performed pitches, dynamics, timbre, and timing of the instruments as well as the pedals, knobs, accelerometers and other physical sources. Additionally, all internally generated signals and timing events are available as sources. These include outputs from all note generators, beat trackers, and ramp generators. The timing information from any repetitive event, loop, delay line, or melodic segment is also available. Destination processes can be triggered by length or the number of repetitions of the phrases occurring in these repetitive events. Triggers are also issued at repetition rates and can be used for advancing or synchronizing events.

All trigger and value sources pass through a flexible router that imparts a unique name and number to the source and allows the source to enter into the matrix upon demand. For the purpose of CPU efficiency, a source and its generative process are halted and the data not sent unless that source is requested by a destination.

Creating a well-behaved modulation manager that is graceful during changes and non-disruptive when idled or activated was essential. The TrigLine module is used throughout the system for conditional control of events and values. A pair of conditions (selected from the Value and Trig sources) is evaluated and then allows other processes to occur.

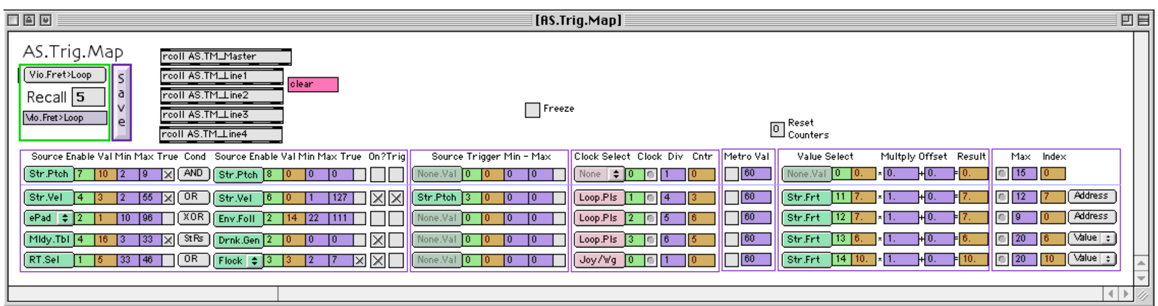


Figure 1: "TrigLine" Conditional System

All modulations sources can be scaled, offset or counted by the destination. This allows detailed tuning in a storable and namable module that is consistent across the system.

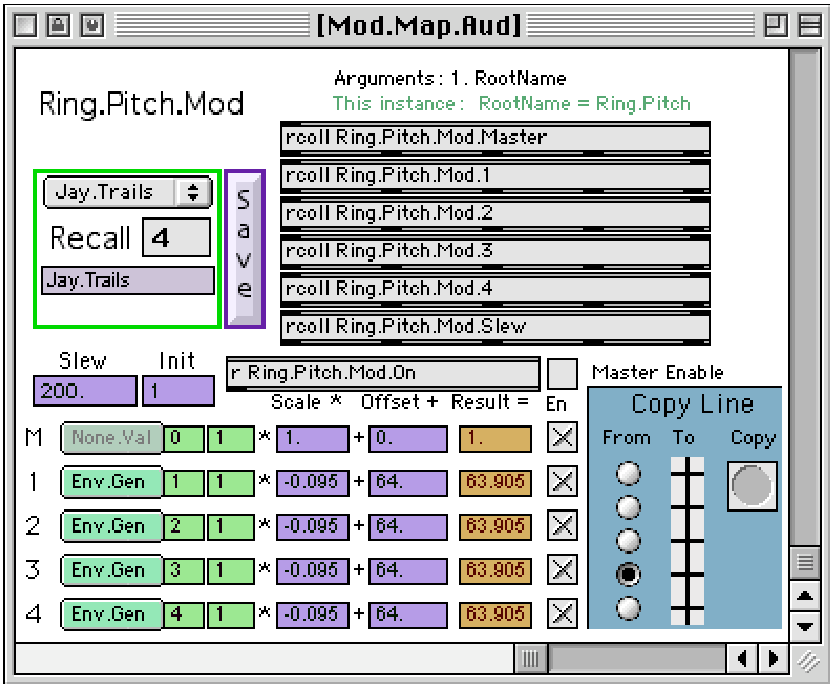


Figure 2: Modulator Scaling

Transport, Timing and Trigmaps.

Once there was a significant body of processes, note generators, and audio effects their reactive control became the dominant research issue. While it is great to have the violin's pitch teach a Markov chain, you may only want it to occur at a certain time. Having the loop recorder grab prime ratios of timed audio is exciting but deciding when to record and play still has to be determined. There were early experiments with the Timeline object, but ultimately this proved too rigid for what was hopefully a reactive control structure.

What evolved is a series of transport controls that are all similar in spirit. In general, pairs of sources and ranges enter as conditional terms to determine if the remaining events are to be evaluated. Upon the conditional requirement's satisfaction, an action can occur which is derived from any modulation source or trigger. For example, if the pitches of the violin and the guitar are within a selected range, then the audio of the bass will be ramped from –40dB to 0dB over 2500ms. After that it could be recorded into the looper until the next seven violin notes are played, then ramp back down to –40dB and stop recording. A comprehensive selection of options and variations are readily available from pull down menus and ranging/scaling parameters.

Any and all functions that can be started and stopped have an associated transport control within TrigLine. Each of these controls can be stored, named and recalled from other control elements within the system. Playback is equally well supported. The above captured loop can sit idle waiting for a trigger such as a cue from the notation system, then play for any arbitrary number of times before smoothly fading out.

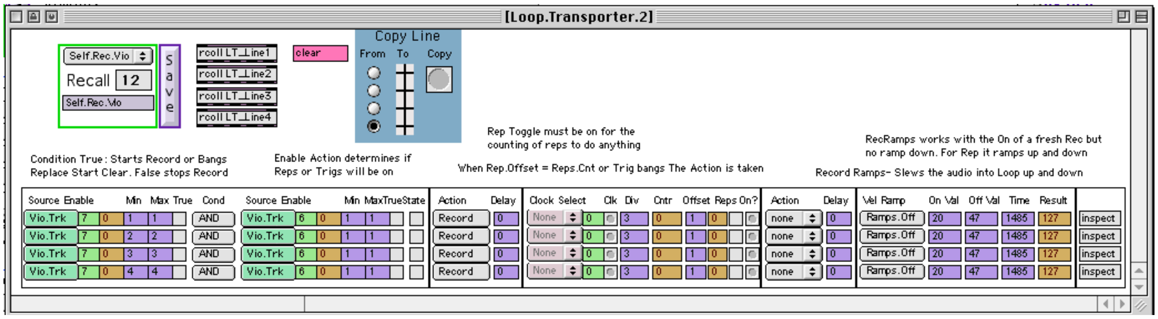
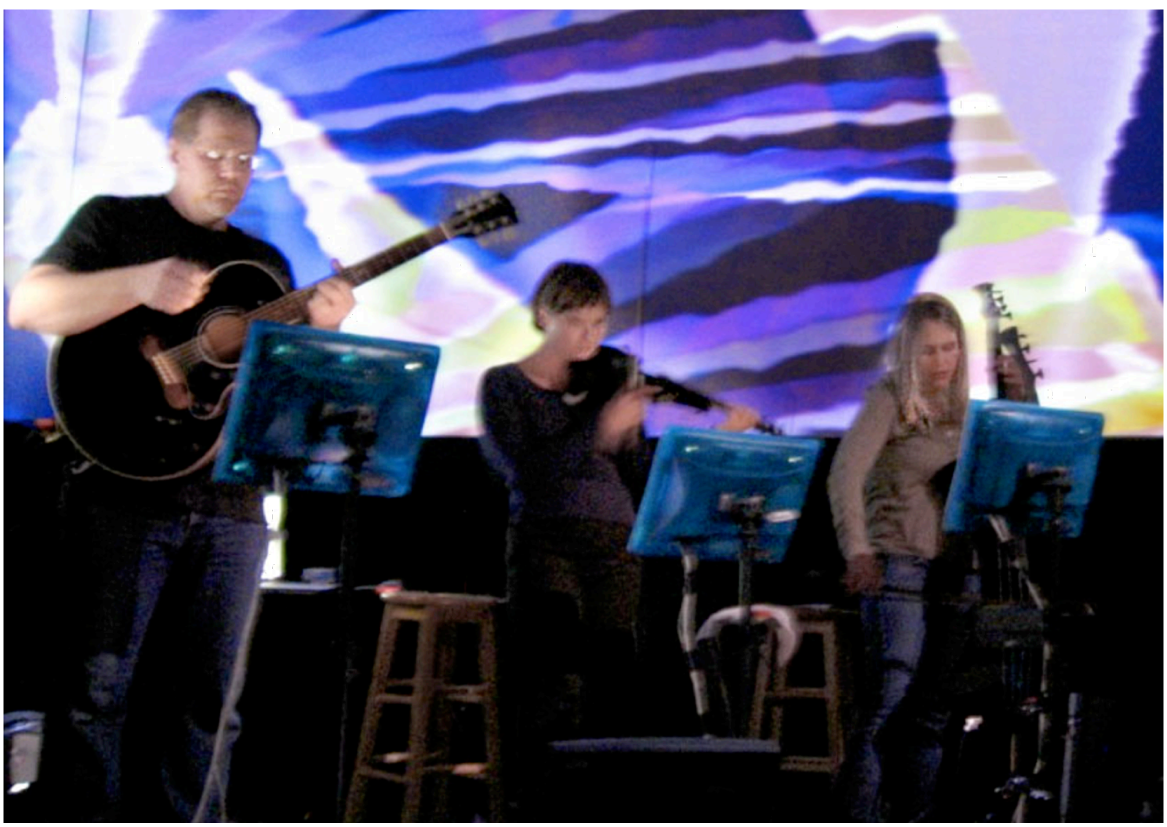


Figure 3: Loop Transport TrigLine

In the module named Master.Slate the time of section (in milliseconds) and section number are generated and made available for each component of a composition. By testing for a specific time in a specific section you can have an absolutely deterministic set of events in temporal sequence. This is the functional equivalent of a Timeline but with easily stretched intervals and conditional occurrences. The price for this flexibility is distributed control. Multiple smaller events are embedded in locations that need to be managed.



TrioMetrik performing at Recombinant Media Labs, SF CA

Figure 4: The Master Slate

Similar functionality is integrated into the TrigMaps for generative systems. Triggers and values can be mapped and recalled for all note and event generators. i.e. Random seeds can be driven by bow angle and incremented by triggers from the bass, but only when the guitar is playing. Initially daunting, these autonomous functions breathe much life and excitement into a piece. When the state and gestures of your fellow musicians determine your timbre and trigger targets, a greater level of listening and mindfulness of interaction is impressed upon the players.

Data Structures and System State Management

As the system grew, more and more time, code and thought went into data management. Ultimately a large set of referable colls (or rcolls) married to a GUI that controlled writing, naming, recalling and messaging was developed. This system allows the abstraction of the musical control data from the inner workings of the program, and enables the stored preset data to be immediately viewable and editable with a standard text editor. A hierarchical tree structure is employed to organize the data from the over 250 user interface windows in the system. Many of these windows have multiple recallable parameter sets. Each of these sets can be named and grouped into a larger rcoll as part of this functional tree. Reuse is supported and encouraged and over time the growing library of named presets have become a welcome resource and time saver.

At the top of the hierarchical control structure is the Master.Slate module. TrioMetrik's entire repertoire is contained and immediately available from this screen. Once a piece is selected by name from a pull down menu the musicians need never touch the computer keyboard. Below the Master.Slate module, there are six main composer UI Screens. Each of these deals with a major characteristic of structure, timbre, or representation that is ultimately controlled by the Master.Slate screen as a named section of a piece. Progress through the sections can be based on events or time or manually controlled. Finer grain changes in each of the subs are controlled by processes within that sub or referred to by another sub's state.

One observation of current computer music performances is the low dimensionality of change engendered in any given piece. People usually come up with some set of connections and then "play the patch". The difficulty of having large sweeping changes along with a subtle control of nuance is clear to anyone who has worked in the field. Easily half of the effort and code of MACIAS has been focused on solving this concern.

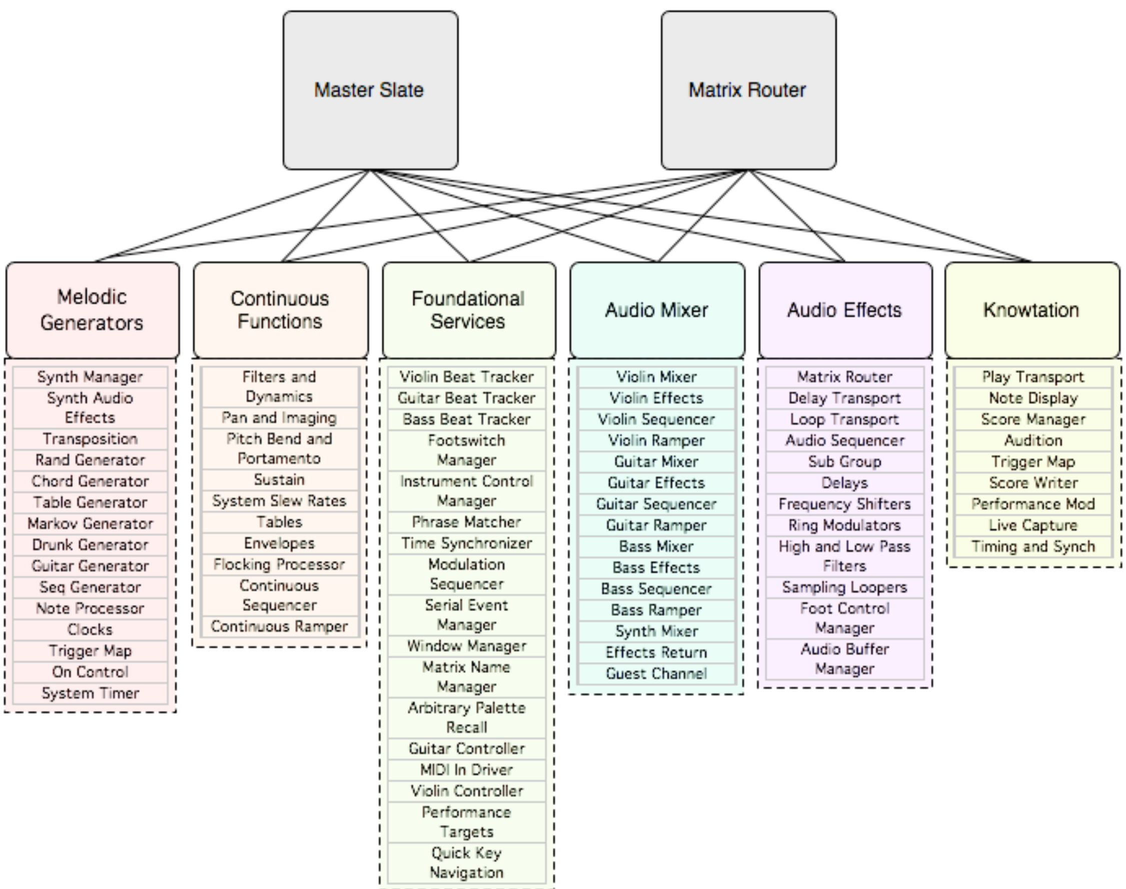


Figure 5: Hierarchy of MACIAS Modules

Musician's Screen and Knowtation

The need to complete the feedback loop from system to musician prompted the Musician's Screen to appear in 1997. As the number and complexity of the system's processes increased, the need for the musicians to be able to quickly monitor and react to these processes became necessary. Concern about the overwhelming amount of information traveling through the system at any given moment led to a separate unique display for each musician, only displaying that musician's currently relevant musical data. However, over time this became both difficult to maintain and unnecessary. A new system was developed using a single interface image distributed to each musician display, and containing all of the information for the entire ensemble. The original worry about data overload proved to be unfounded. Musicians are accustomed to playing from multi-instrumental scores and filtering out the information that is not relevant to their performance. With practice, and knowledge of the interface's structure and indicators, reading the display becomes transparent. In the event that a different set of information is needed to accommodate a specific piece, the display modules are malleable and re-locatable depending on the processes in use, relieving the pixel real estate issue.

Several attempts at novel ways of representing notation for musicians made it apparent how deeply ingrained and workable the present folded paper system remains. Early attempts at showing just the current note in graphical form did not allow the musicians to anticipate the score, and so was unworkable in performance. Similarly, piano scroll was not learnable for live performance. The present scheme (termed Knowtation) presents two staffs to the performer that page alternately (based upon an external from Peter Swinnen extended by the author and Richard Dudas). This allows the musician to see the phrase with some completeness and to look ahead in preparation for next actions. The left to right paging is comfortable and allows painting of up to 16 events per page. A tablature version (also representing 16 events over 2 pages) for the guitarist provides greater density and ease of play. Some simplifications in notation have been accepted to make the representation scheme more flexible. There are no time or key signatures or bar lines, and each note is played for its own common time value.

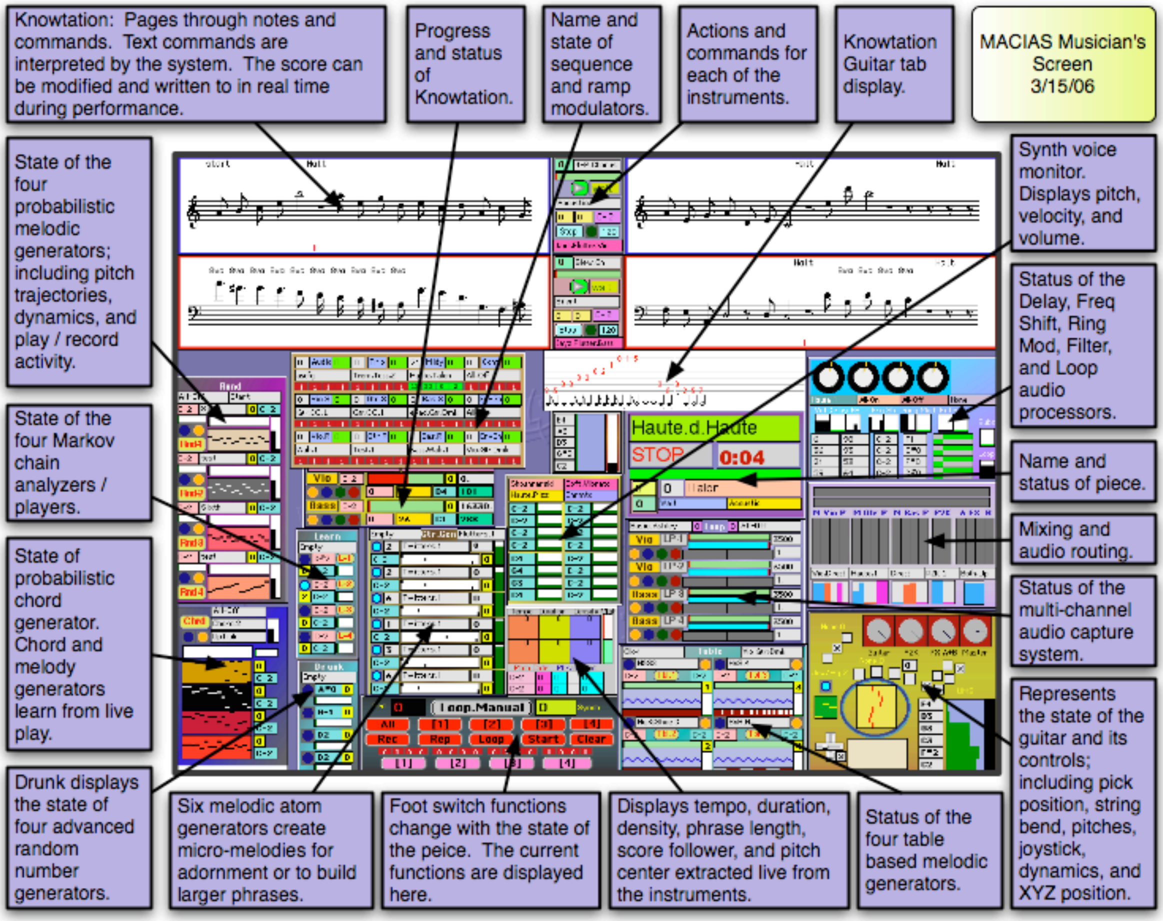


Figure 6: Sections of the Musician's Screen

Several modes of motion through the score are supported as well. Play mode follows normal elapsed time at a selectable tempo. An indicator below the staff shows current time location. Step mode allows an external trigger to move to the next note in the phrase. Page mode presents a motif for the musician to play freely for some extended time period. An external event will advance to the next motif.

Text based commands can be embedded into the score to send signals into the modulation matrix. These can start a loop recording or playing, change an instruments timbre or modify the score itself. Text messages to the musicians can also be embedded in the score.

Pitches to be performed can come from a score or be generated by other processes or musicians during a performance. Key and mode changes can be applied. Automatic ranging and wraparound guarantee the represented notes fall within the range of the target instrument.

Modulation Interfaces

With the large quantity and variety of modulation sources and destinations available several types of modulation interfaces evolved to control the modulation data. These are available distributed throughout the system and are usually associated with larger functional modules, i.e. there are two associated with each of the instrument's input audio.

The first of these was modeled on the old style 16-step analog sequencer. Here four sets of up to sixteen values can be stored and recalled. A TrigMap is used to step through each or all of the four lines of values. Then each of the values is sent to the target destination. This approach allows an easily modifiable set of values that encourages experimentation. Targets include the preset numbers of other modules as well as typical control parameters such as filter cutoffs and levels.

For example, each of the four lines of sixteen values can be mapped to the RingMod drive index. The step position is set by the semitone value of each of four strings of the violin, so the E string open selects position one, the F on the same string selects position two, etc. The value of each step is set so that higher modulation indices are determined by the position each pitch has in a selected scale. So octaves, 5ths, and 3rds can have a low RingMod drive but passing tones can be highly modulated.

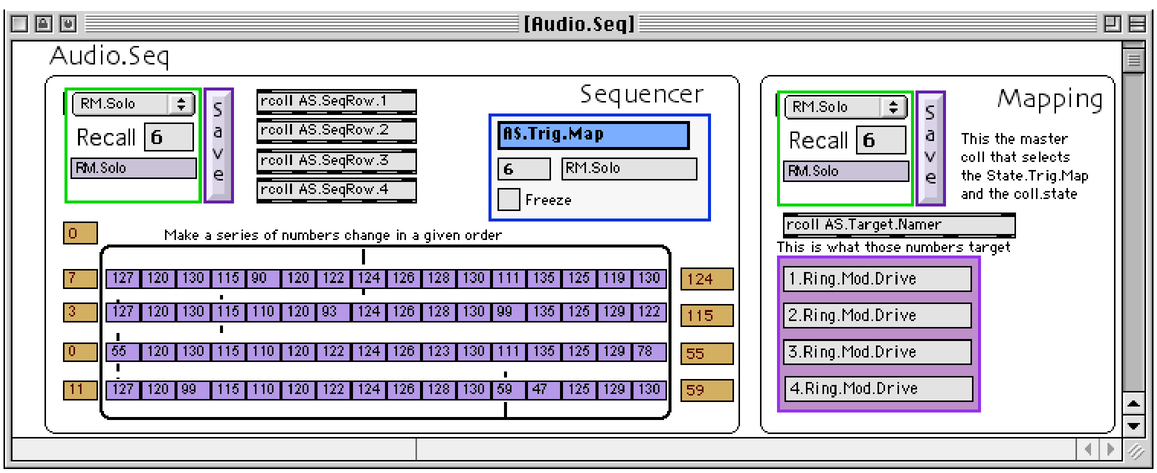


Figure 7: AnaSeq; A Modulation Interface

For complex timbre modulation the Ramper module provides a mapping of four control values through four multipoint line segments routed to sixteen scale & offset modifiers and then to sixteen selectable targets.

One example uses the state of an instrument's past performance to determine its current sound. As the number of notes played on a given string increases, each of the corresponding four line segments advances and the results are mapped to a dozen timbre modifiers (bass, pan, distortion, reverb, delay, pitch-shift, etc).

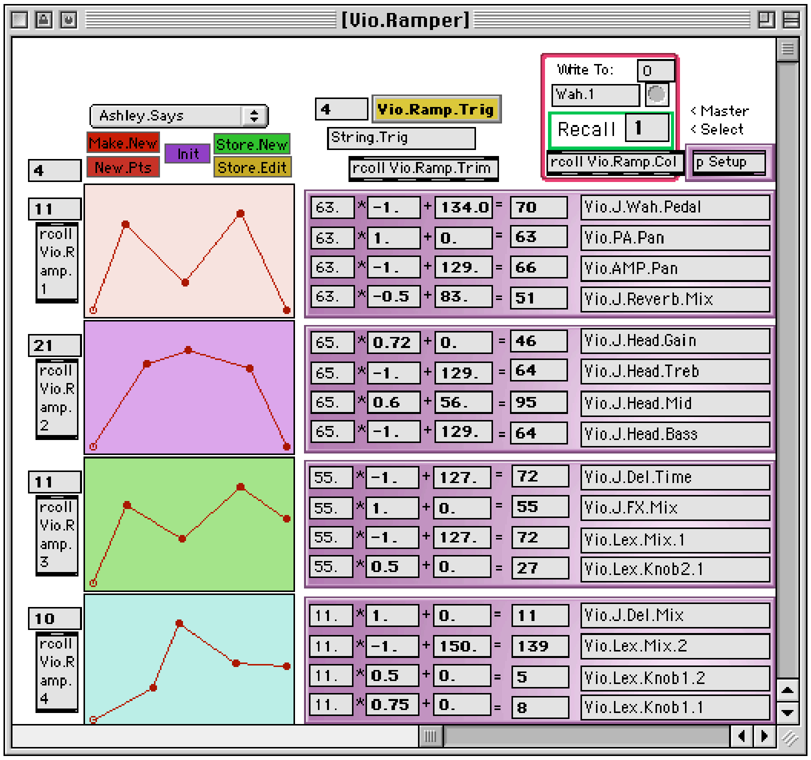


Figure 8: Ramper; Another Modulation Interface

Note and Event Generators

Techniques for the generation and manipulation of pitches, values, and events are a requirement for many new compositions. MACIAS contains a large collection of stylized generators that can be interconnected at numerous functional levels providing a significant series of options for the composer. Four to six instances of each of the following are always instantiated and ready for use.

- **Rand.Gen** – Probabilistic players that can recall a named state and have values written to its variables in real time.
- **Chord.Gen** – Four note chord generator similar to the above but designed for the manipulation of multiple parallel values best suited for chords.
- **Table.Gen** – Table based players where pitch, velocity, timing, duration and delay are represented in editable tables. A separate utility allows the analysis (both real-time and pre-processed) of MIDI files or live streams for representation in this more malleable format.
- **Learn.Gen** – 1st order Markov Chains that learn from live or generated input and are coupled with a flexible player.
- **Drnk.Gen** – Seedable Random Number Generators with presettable ranging, step size and response characteristics.
- **Gtr.Gen** –Triad Generators, six sets of 16 triads can be mapped and elicited by any set of values and events. Great for ornamentation or autonomous note clusters.

- **Seq.Gen** - MIDI Seq players for those who simply must play a sequence. Tempo, step/quantized, scale and transpose expand upon basic playback.
- **Note Processors** – Harmonic & Rhythmic post processing for all events include preset and performable transposition, duration, timing, harmonic filtering, and pitch interpolation methods.
- **Preset, continuous controller & effects** management for hardware synthesizers contain an exhaustive implementation of all target variables for the external synthesizer/sampler. These include filter cutoff and Q, VCA attack and release multipliers, level, pan, (absolute as well as movement rate and depth), bend, sustain, portamento, etc. All external parameters are abstracted to appear synonymous with all internal modulation targets.

Audio Routing and Effects

As discussed earlier extensive signal processing is performed on the instruments using an array of external processors that are tightly controlled from within the program. A pair of stereo feeds (mono processed and polyphonic processed) from each of the three instruments enters the CPU through two Firewire audio interfaces. These signals enter the Mixer function where a final three band EQ can be applied to each of the feeds before they are routed to the PA, to 2 internal effects bus inputs, and individual instrument amps or out for further localization.

As in a large modular synthesizer, certain functions are here considered basic and essential to live performance. Four instances of each of the following processes (20 internal effects blocks) are always available:

- **Looping Delays** – Stereo delays with time settable from any set of trigger events or values. Inputs and outputs can be automatically ramped or gated based upon selections from the Modulation Matrix. Feedback and freeze allow layering of tracks.
- **Frequency Shifters** – Separate control for frequency and drive as well as modulation setup.
- **Ring Modulators** - Separate control for frequency and drive as well as modulation setup.
- **Hi and Lo Pass Filters** – Filters can be cascaded for series and parallel operation.
- **Sampling Delays** – Live audio can be sampled and manipulated in these arbitrary length buffers. Times and rates can be synchronized from any of the modulation sources. Completely deglitched, all parameter can be manipulated with impunity.

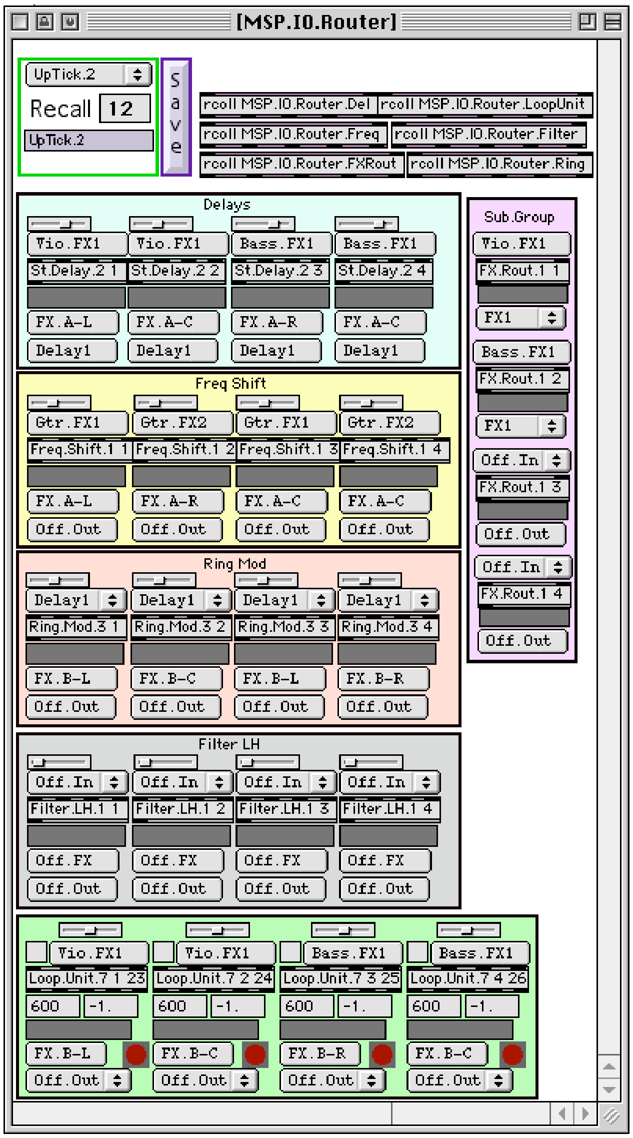


Figure 9: IO Router

Audio sources include the instruments, synthesized tones, and a stereo guest channel. All effects outputs can be cross-routed to any effect input for ease of chaining. If an effect has no input selected it is turned off thereby saving cycles. These routings can be named and recalled.

A goal of the system was live use with no soundman. Loudness management became a major concern as the number of audio generators grew to 52 voices from live, looped, processed and synthetic sources. This prompted the founding of Octiv, Inc. and the creation of ultra efficient multiband dynamics processors. [5]

Sub-groups of audio are bussed and moderated by these OMX dynamics processors that are now part of the standard Max/MSP release. In addition to controlling peaks and levels, a spectral “fingerprint” can be assigned to each of the groupings that have added greatly to the ability to fuse these sets of sounds.

Prior to the invention and addition of the OMX processors, it was very difficult to get a good ensemble result from the disparate audio contributors. We all recognize this when a live musician plays over a recording. We know they are separate events. Overcoming this phenomenon has been a major research project tied to the tuning of the system.

Video Processing

With many aspects of a performance already abstracted into values and events the temptation to use these controls to manipulate video during the performance was hard to resist. Extending the modulation parameters using UDP to a dedicated video PC running Jitter enables the responsive control and processing of imagery.

Certain mappings are obvious and quite satisfying. Advancing a movie at some multiple of the beat creates an immediate link for the viewer/listener. Mapping the zoom of an image to the strength of a beat further enhances this result. Distorting an image via a texture map and controlling the amount of the effect by inverse importance of scale degree creates a subtler result. This area is a new but exciting extension to the MACIAS system.

Modes of Mind

After extensive examination and experimentation I have come to realize there are three modes of operation when it comes to working with electronic music; tool building, composing, performing. One goal of the system has been to keep these different disciplines separated in approach and function.

Tool building encompasses all aspects of design and implementation. This includes hardware, programming, testing and system tuning. Patches are unlocked and modified and instruments are open and reworked. This is definitely the programmer's mind.

Composer's mind is the imbuing of an idea into the system by making choices and defining ranges and relationships. Once a piece is named in the composer's screen, you never have to open or edit a patch to fully define any and all aspects of the composition. Resources are well behaved and readily available. Transitions have been pre-examined for consistency and glitch free operation. As a composer in this system you can focus on the realization and refinement of your ideas without resorting to further programming.

In performance there is no contact with the keyboard after selecting from a pull down menu the name of the piece to be performed. All input comes from the musicians or from the conditional and temporal state of the composition. All necessary visual data is presented to the musician's screen. The performers are free to perform.

Tools and what they produce

Advanced tools make it easier and are often required to create refined art. Think of Frank O. Gehry's work in architecture.[6] Software tools that grew from shipbuilding and aircraft design enable him to explore novel buildings techniques and original construction methods. The ability to quickly visualize and specify innovative structures and methods makes the artist ask larger and more ambitious questions. This all flows from working with powerful and refined instruments.

The inverse is also true. If it takes two months to realize an idea or hear the result of an experiment one can become bound to the concept through sheer momentum. We become attached to our ideas by the pure investment of time and energy. Good tools save time and energy. Good instruments inspire. A cigar box, broomstick and rubber bands can meet the basic requirements of a violin, but the lasting beauty of an instrument lies in its reworked evolution to yield a true creative enabler instead of a simple proof of concept.

A goal of this system was to make it as endearing as a favorite violin or the 9 foot grand at your mother's house. After years of work, and now years of play, I am pleased to report there are many such moments.

Thanks to Barry Threw, Ashley Adams, Marielle Jakobsons, Chris Muir, and Richard Dudas.

---

[1] For a more detailed history visit "http://www.beamfoundation.org/technology/history".
[2] String instruments have been the subject of much research in the computer music community. In particular, see Tod Machover's HyperInstruments, "http://www.media.mit.edu/hyperins/".
[3] http://www.zetamusic.com
[4] For a complete list of modulation destinations and sources, see "http://www.beamfoundation.org/downloads/ModList.pdf".
[5] Octiv, Inc. was sold to Plantronics, "http://www.octiv.com".
[6] http://en.wikipedia.org/wiki/Frank_Gehry

Further Reading:

BEAM Foundation. (2006). BEAM: Music For A New Reason. http://www.beamfoundation.org

Chamberlin, H. (1985). *Musical Applications of Microprocessors*. Hayden Book Co.

Cycling '74. (2006). Cycling '74 || Tools For New Media. http://www.cycling74.com

Kurzweil, R. (2000). *The Age of Spiritual Machines: When Computers Exceed Human Intelligence*. Penguin (Non-Classics).

Machover, T. (1992a). *Hyperinstruments: A progress report, 1987-1991*. MIT Media Laboratory.

Machover, T. (2006b). Dreaming a New Music. *Chamber Music*, *23*(5), 46 -54.

MIT Hyperinstruments. (2006) Hyperinstrument Homepage. http://www.media.mit.edu/hyperins/

Rowe, R. (2004). *Machine Musicianship*. The MIT Press.

Strange, A. (1972). *Electronic music;: Systems, techniques, and controls*. W. C. Brown Co.

Winkler, T. (2001). *Composing Interactive Music: Techniques and Ideas Using Max*. The MIT Press.

Zeta Music, Inc. (2006). Zeta Music MIDI electric violins, violas, cellos and basses. http://www.zetamusic.com