MARS = NERGAL + ARES
P. Andrenacci, F. Armani, A. Paladin, A. Prestigiacomo, C. Rosati, S. Sapir, M. Vetuschi

To cite this version:
P. Andrenacci, F. Armani, A. Paladin, A. Prestigiacomo, C. Rosati, et al.. MARS = NERGAL + ARES. Journées d’Informatique Musicale, 1997, Lyon, France. hal-03106728

HAL Id: hal-03106728
https://hal.science/hal-03106728
Submitted on 12 Jan 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
**MARS = NERGAL + ARES**

P. Andrenacci, F. Armani, A. Paladin, A. Prestigiacomo, C. Rosati, S. Sapir, M. Vetuschi  
IRIS s.r.l. Parco La Selva I-03018 Paliano Italy, mc2842@mclink.it

**Abstract:** The Musical Audio Research Station (MARS) is a programmable specialized digital machine for real-time audio applications which has been entirely developed by the Italian Bontempi-Farfisa research institute IRIS. The MARS workstation is composed of:

- one or more sound boards, called NERGAL connected to one or more audio units;
- a PC running the graphical interactive development environment called ARES.

The first version of MARS, which has been presented at the ICMC92 was running on Atari computers. In this paper IRIS will present improvements and new features of the new version of MARS which has been entirely redesigned for the PC world.

1. **MARS overview**

MARS derives from the MUSICn philosophy, as it maintains the basic approach developed by M. Mathews [Mathews and al. 1969]. Synthesis algorithms are specified as combinations of building blocks; the activation of instruments is controlled by note events. Its historical background can also be found in the main features of the 4X system [Favreau and al. 1986] (high performance real-time sound processing and interactivity) and in GROOVE general concepts for controlling note parameters through functions of time during performance [Mathews, Moore 1970]. It emphasizes symbolic representations of data in order to offer a user interface which is independent of hardware architecture and which makes musical applications portable.

1.1 **Architecture**

MARS is a traditional three level based (DSP, controller and host) system. The IRIS X20 Digital Signal Processor is dedicated to real-time sound processing [Cavaliere and al. 1992]. The Motorola MC68302 microcontroller supports real-time operating system RT20M [Andrenacci and al. 1992], developed for sound generation board NERGAL. The host computer supports graphical development environment ARES. This architecture guarantees high performance and gives the opportunity to distinguish between sound processing and performance controls, also allowing musicians to use MARS as a MIDI instrument.

1.2 **Objects**

MARS has a predefined set of primitive objects which can be edited using ARES graphical environment to build more complex objects such as algorithms, tones, orchestras and MIDI performance setups. Following are definitions of the main objects managed by the system:

The MARS basic audio objects are user patchable algorithms which correspond to specific signal processing operations (microcode for the X20s). Algorithms are built graphically and are immediately implemented on the sound generation board avoiding compilation time.

For each algorithm, the user can configure a number of tones. A tone defines for all algorithm parameters the rules for calculating their values and their relationship with MIDI controls and events.

An orchestra is a collection of polyphonic algorithms (clones) and a map of their connection with audio busses, DACs and ADCs. For each orchestra, the user can create a number of tone maps which define the collection of tones to be played and the overall orchestral MIDI environment, also called performance environment.

The creation and editing phases of these objects involve different contexts which are managed by different processors. For example, the output data of an algorithm or an orchestra editing session involves the DSP level (X20 microprograms, data memory values, sample memory, ...), while the output data of a tone or a tone map editing session involve the control level (i.e. the operating system running on the

---

1 This title has been borrowed from a paper written by Ralph Winrich you can find on the WWW at http://spacelink.msfc.nasa.gov/Instructional.Materials/Curriculum.Materials/Sciences/Astronomy/our.Solar.System/Mars/Mars.Activities. In this paper R. Winrich explains the origins of the Mars name and many other interesting features about the planet.
microcontroller) which manages the MIDI messages and the performance environment interactions (see Figure 1).

Figure 1: MARS objects hierarchy

2. Description of the new hardware

The minimum MARS hardware requires the sound generation board NERGAL with 8 megabytes of sample RAM and a low-cost audio unit. Hardware expansion allows to plug up to 8 sound generation boards together into the ISA bus extensions of the host PC. On each board, sample RAM may be expanded up to 64 megabytes. Furthermore, IRIS provides a configurable professional audio unit which allows up to 8 analog in and 8 analog out, and 8 AES-EBU in and 8 AES-EBU out or another digital audio interface.

2.1 NERGAL, the new MARS ISA sound board

The sound generation board is a full size standard IBM ISA card running at a sampling rate of 44.1 kHz. The board may be divided into three main sections: DSP, control and mixer sections.

The DSP section contains two dedicated IRIS DSPs called X20 with their respective program and sample memories. These DSPs are memory mapped onto the microcontroller memory area. The X20 is a fully programmable DSP. Its architecture is highly parallel and allows pipelined operations. Its internal arithmetic is 2’s complement fixed point with a 16 x 16 bit multiplier and a 24 bit arithmetic and logic unit.

Each X20 has two blocks of internal data memory which may be addressed in parallel. It also contains a 512 x 64 bit words of external program memory, and may access to external sample memory. The sample memory is supported by SIMM modules with 4 up to 32 megabytes of DRAM.

Each X20 provides four audio input channels and four audio output channels. The audio signals are managed on 16 bits. The X20s may be used in parallel or in serie, by sending the four outputs of the first one to the four inputs of the second one.

The control section is managed by a proprietary real-time operating system, called RT20M, running on a MOTOROLA MC68302 microcontroller. The RT20M manages all the MARS objects described previously, and all their interactions with the DSP section, the mixer section and the external world. It also controls any communication through the three ports available on the board: MIDI, RS232, and a special parallel one for high speed connection to the PC’s ISA bus.

The mixer section controls a set of predefined audio mixing operations among boards and audio units.
Each board have one connector to/from the audio-unit, and four connectors (two inputs and two outputs) which allow multi-board links. This last feature allows to create a MARS boards network with a parallel, a serial or a mixed configuration using one or more audio units.

### 2.2 MARS Audio units

Two types of IRIS audio units are available for NERGAL: low-cost and professional.

The low-cost unit is a small external box (1/3U) with two monophonic input lines, four monophonic output lines, and one stereophonic headphone output.

The professional audio unit is a 2U standard rack module. It is modular and may pilot up to 8 input/output analog channels and 8 input/output digital channels. Each unit can be shared between two different NERGAL boards.

### 3. Description of MARS new software ARES

The MARS music workstation is an open system that needs to be programmed and configured. Such operations are all managed by the proprietary development system, the IRIS Audio Resource Editing System (ARES) running on Windows 3.x and Windows 95. ARES hides all the low-level feature of the system, so users don’t have to know anything about the X20 architecture, or about the RT20M details. Furthermore, ARES provides a set of default behaviours which help users to quickly program the system. Finally, ARES provides any editing in a real-time interactive context.

#### 3.1 Use and programming

The work on MARS usually develops in three steps [Favreau, Sapir 1993]. The steps order described in the following paragraphs is not a strict necessity, on the contrary many editing steps can be done in parallel thus increasing the system flexibility.

The first step is to create and debug algorithms with the algorithm editor [Armani and al. 1992] for the processing of signals: synthesis, analysis and effects. This system provides the user with a rich and expandible library of patchable modules for signal processing and with a series of debugging tools (oscilloscope, VU-meters, etc.) which simplifies the creation and debugging of algorithms. Furthermore, the system implements the modules in real-time, as soon as they are installed by the user. This operation requires no pre-processing or compilation of the algorithms, thus allowing one to instantly hear the resulting sound.

The second step is to create and debug timbres [Palmieri, Sapir 1992] with the tone editor: define their values and the rules for calculating parameters of previously defined algorithms. Any algorithm can be configured with a number of tones (defining its parameters values, its envelopes and its waveforms, and assigning and defining MIDI control parameters, etc.) by means of a series of secondary editors and graphically interactive tools that are all part of the ARES package.

The third step is to create and debug the orchestra with the orchestra and the tone map editors: listing the algorithms and timbres which will be used and defining the MIDI performance environment associated with the orchestra. While developing new algorithms and timbres, the user creates a sort of database of sound resources that can be later used inside one or more orchestras for the execution of musical pieces. These orchestrations can be considered true performance environments once the user specifies the polyphony of various algorithms used, the list of timbres to be played, the assignment of MIDI channels, and finally the configuration of the audio bus.

Once these three phases are completed, MARS can be used for live music by connecting it to a gesture MIDI device or some MAX application [Puckette 1991], by playing a MIDI score executed by a sequencer, or by combining these techniques.
3.2 High-level Interface and user friendly interaction

ARES hides from the user all low-level implementation and communication aspects related to MARS sound generation board. Multi document metaphor for all MARS objects editors, and real multi-window management makes editing an easy task (see some windows examples on Figure 2).

![Figure 2: ARES windows examples](image)

Design, enhanced graphics, intensive use of short-cut keys, tool bar buttons, and information displayed in the active document status bar help users to interact with ARES.

In general you can act on any kind of object by following these two steps: selecting the object by clicking on it, and carrying out an edit operation by using the mouse, a short-cut or a menu command. ARES On-line Help allows user to reach any kind of information on patchable modules, glossary terms, editors layouts and commands, procedures and so on. Furthermore IRIS provides a complete tutorial and a documented set of demo which enable beginners to easily start to work with the MARS workstation.

3.3 A browser to navigate through MARS database and other tools

While developing MARS objects, the user creates a sort of database, the catalog of sound resources. With ARES, objects can be accessed by using the Catalog Browser. The Catalog Browser hides all ARES subsidiary files focusing only on ARES object names. ARES also provides a set of software tools which improves MARS overall performance.

- The oscilloscope allows users to display audio signals on four independent tracks, picking it up at any point of patch data flow.
- The MIDI keyboard allows users to send standard channel MIDI messages through one of the available communication ports.
- The wave table manager allows users to manage MARS sample memory. ARES supports all major file formats including binary Intel, and binary Motorola.
The juke box allows users to send to a board one or more performance environments in a batch session without running ARES. Furthermore, the juke box allows playing MIDI files which makes it very useful for stage use of MARS.

4. Some examples of DSP applications

MARS is a development system for interactive desing and implementation of digital audio algorithms particularly oriented to musical applications. Any kind of real-time digital signal processing techniques such as analysis, synthesis, any kind of filters, and sound effects may be realized.

4.1 A richer set of proprietary algorithm modules

ARES provides about 200 modules to be used in the algorithm editor environment. This library covers low and high level needs for DSP applications such as arithmetic and logical operations, many types of oscillators and complex sound generators, physical modeling building blocks, and audio effects. The DSP modules are logically grouped in families as described in Table 1.

<table>
<thead>
<tr>
<th>Families</th>
<th>Modules</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic operators</td>
<td>adder, multiplier, ...</td>
</tr>
<tr>
<td>Sample memory access</td>
<td>read table, write table, ...</td>
</tr>
<tr>
<td>Oscillators</td>
<td>simple oscillators, PCM oscillators, ...</td>
</tr>
<tr>
<td>Random generators</td>
<td>random, 1/f noise, noise exciter ...</td>
</tr>
<tr>
<td>Undersampled generators</td>
<td>envelope, LFO, vibrato, ...</td>
</tr>
<tr>
<td>Modulators and distortors</td>
<td>different FM, AM, ring modulation, ...</td>
</tr>
<tr>
<td>Complex generators</td>
<td>pulse generator, additive generator, ...</td>
</tr>
<tr>
<td>Logical operators</td>
<td>inverter, clipper, max, min, ...</td>
</tr>
<tr>
<td>Logical transformers</td>
<td>triggers, multiplexers, noise gate, limiter, ...</td>
</tr>
<tr>
<td>Time transformers</td>
<td>delay lines, ...</td>
</tr>
<tr>
<td>Spectral transformers</td>
<td>various types of filters</td>
</tr>
<tr>
<td>Pitch transformers</td>
<td>harmonizer, pitch shifter, doppler effect, ...</td>
</tr>
<tr>
<td>Amplitude transformers</td>
<td>mixer, panpot effect, envelope follower, ...</td>
</tr>
<tr>
<td>Physical models</td>
<td>strings, reeds, tubes, masses, spring, ...</td>
</tr>
<tr>
<td>Effects</td>
<td>echo, flanger, chorus, reverb, exciter, ...</td>
</tr>
<tr>
<td>Miscellaneous</td>
<td>parameters, debugging tools, audio i/o</td>
</tr>
</tbody>
</table>

Table 1: ARES proprietary modules list

The modules may be easily picked-up from the families palette shown in Figure 3 and patched in the ARES algorithm editor. In the future, while this modules list will constantly be updated and powered by IRIS, users will have the means of easily organizing their own lists, building their own modules set and editing their families palette.

![Figure 3: ARES families palette](image)

4.2 Algorithm and orchestra development examples

In the example displayed on Figure 4 we can see, on the left, an algorithm which uses physical modeling modules to simulate a clarinet. The clarinet model uses the classic excitation/resonator model: a static reed followed by a pipe. The pipe is simple, it is composed by a piece of tube, a hole and a bell. The output is given by mixing the signals produced by the four elements.

On the right part of the Figure 4 we can see an orchestra example. The orchestra is composed by 4 algorithm families as described in the Table 2.

![Figure 4: Algorithm and orchestra development examples](image)
Table 2: Orchestra components

The communication between algorithms and between audio I/O is achieved by the orchestra busses. These busses allow the accumulation of audio signals, but also feedback effects. For example, a feedback effect is obtained by sending the output of the string algorithm family to the input of a distortion effect. Then the output of the distortion is sent back to the string in order to be added to the plucked excitation of the string model.

4.3 Debugging tools

In the example displayed on Figure 5 we can see, on the left, the ARES oscilloscope window during an algorithm debugging session. The first track represents an envelope signal, while the second one represents an enveloped waveform signal.

On the right, we can see the ARES watchers window which display some algorithm’s data with VU-meter and numerical representations.
4.4 Tones and MIDI performance environment configuration

MARS is also a development system for sounds and MIDI environment that allows musicians to use it as a musical instrument, once configured, such as any MIDI equipment of a musical studio. When editing a tone, ARES provides secondary graphic editors which allow users to define their tables, envelopes, LFOs, and other algorithm parameters. In the example displayed on Figure 6 we can see, on the left, the ARES dynamic parameter window which allows editing the relationship between any scalar parameter of an algorithm with any MIDI devices.

![Parameter to MIDI mapping and tone examples](image)

In this example the freq_2 parameter is mapped onto two MIDI devices: the discrete Key value of the MIDI Note On message, and the continuous Pitch Bend control. The MIDI values are transformed by using two conversion tables (hztemp and pitchb) and a formula which allows the following operation:

\[
\text{freq}_2 = \text{hztemp}[\text{Key} + 12] \cdot (1 + \text{pitchb}[\text{Pitch Bend}])
\]

On the right, we can see the ARES tone window which lists all of the algorithm’s entry points: envelopes, different kind of scalar parameters and LFOs.

Finally to complete the performance environment definition, users have to list the orchestra’s tones and the current tone to MIDI channel mapping by using the tone map editor, and to load all the samples and wave tables in the NERGAL’s sample memory by using the wave table manager tool.

5. Extending MARS with new applications

MARS remains an open system. It is possible, as for the Atari version, to expand MARS possibilities, or to specialize it for particular needs, by programming dedicated new applications [Andrenacci and al. 1993], [Armani and al. 1994]. A development toolkit consisting of C/C++ function and class libraries allows programmers to develop their own MARS applications without using ARES. However, these applications will be able to exchange catalogs of sound resources with ARES and to share a common NERGAL network.

6. Conclusion

The new release of the MARS workstation presents a completely new design of the sound board NERGAL and of development environment ARES. The MARS hardware is now directly pluggable into the ISA bus extension of a PC, and allows multi-board configuration to improve the computing power of the workstation. Furthermore, a new family of audio units allows a more flexible configuration of the audio I/O. ARES has an improved user interface, a new design of all editors and tools, and it allows real multi-task operations. It contains a browser which makes MARS objects database navigation easier. A richer
proprietary modules library and the possibility of expanding this library increase the DSP functionalities of the system.

ARES replaces the old EDIT20 package running on the Atari. It is an improved environment with multi-board and multi-task facilities, but the original “philosophy”, and most of the EDIT20 features have remained unchanged to maintain the compatibility and the continuity with old applications. A development toolkit offers programmers the chance to create their own MARS applications and to extend the functionalities of the workstation.

MARS is a flexible musical workstation which is used in a wide range of research, musical and teaching contexts such as universities, music conservatories, computer music research centers, music studios, interactive museums, and in the musical instruments and audio industry.

Acknowledgements
Special thanks to R. Bessegato and P. Pisani who designed the MARS hardware, and to the previous MARS staff who contributed to the overall MARS design.

References