

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

**ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600**

UMI[®]

**An Application of Max/MSP in the Field of
Live Electro-Acoustic Music. A Case Study**

Oles Protsidym

Faculty of Music, McGill University, Montreal

August 1999

**Thesis submitted to the
Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of
Master of Arts in Computer Applications in Music**

© Oles Protsidym, 1999



**National Library
of Canada**

**Acquisitions and
Bibliographic Services**

**395 Wellington Street
Ottawa ON K1A 0N4
Canada**

**Bibliothèque nationale
du Canada**

**Acquisitions et
services bibliographiques**

**395, rue Wellington
Ottawa ON K1A 0N4
Canada**

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-65472-9

Canada

Abstract

For many years the integration of real-time digital audio signal processing with dynamic control has been one of the most significant issues facing live electro-acoustic music. While real-time electroacoustic music systems began to appear as early as the 1970's, for approximately two decades a lack of portability and prohibitive cost restricted their availability to a limited number of electronic music studios and research centers.

The rapid development of computer and music technologies in the 1990's enabled desktop-based real-time digital audio signal processing, making it available for general use. The introduction of the Max/MSP programming language in the 1990's ushered in a new era in electro-acoustic music by bringing a new unified environment for real-time digital signal processing and dynamic control to a wide audience.

The potential of the Max/MSP programming environment for applications to live electro-acoustic music is the subject of this thesis. A prototype example of a real-time electro-acoustic music system based on Max/MSP is developed and discussed.

Sommaire

L'intégration du traitement de signaux audio-numériques en temps réel avec le contrôle dynamique a pendant longtemps constitué un des enjeux les plus significatifs dans le domaine de la musique électro-acoustique "live". Alors que des systèmes électro-acoustiques en temps réel firent leur apparition dès le début des années soixante-dix, leurs problèmes de portabilité, ainsi que leur coût élevé, limitaient leur disponibilité à quelques studios de musique électronique et centres de recherche.

Le développement rapide des technologies informatiques et musicales dans les années quatre-vingt-dix a permis l'avènement des systèmes de traitement "desktop-based" des signaux audio-numériques, les rendant ainsi accessibles à tous les intéressés. L'arrivée du langage de programmation Max/MSP dans les années quatre-vingt-dix inaugura une nouvelle ère dans la musique électro-acoustique, mettant un nouvel environnement de traitement de signaux en temps réel et le contrôle dynamique à la disposition d'un public plus vaste.

Le potentiel de l'environnement de programmation Max/MSP pour les applications à la musique électro-acoustique en direct constitue le sujet de ce mémoire. Un exemple d'un tel système sera présenté et discuté.

Acknowledgements

First of all I would like to express my gratitude to Dr. Bruce Pennycook for his continuous support, understanding and guidance throughout years of my study and research at McGill. His competence, insight, and willingness to explore uncharted territories were always inspiring.

Another person who provided valuable assistance to this work is Mr. Zack Settel, who, despite his busy schedule, was always available and willing to give extremely useful suggestions related to the software design.

I would also like to thank my colleagues, especially Geoff Martin and Elbert McLaughlin, whose presence at McGill enriched my learning environment. Special thanks to Sean Ferguson and Mrs. Anne Holloway for proofreading this text, as well as to Helen M. C. Richard, Mrs. Claire Richard and Mr. Zack Settel for translating the abstract into French.

I would like to acknowledge all of my friends in Lviv and Montreal, for their good will and help, especially Myroslava, whose love and support is always appreciated.

Finally, there are no words to express my thanks to my parents who helped me to understand what is truly valuable in this world and who taught me to find my own path. Their love, understanding and care are beyond imagining.

An Application of Max/MSP in the field of Live Electro-Acoustic Music. A Case Study

Table of Contents

1. Introduction	1
1.1 Thesis Structure	3
2. Real-Time Electro-Acoustic Music Systems	4
2.1 Electro-Acoustic Music	4
2.1.1. General Classification and Selected Definitions	5
2.1.2 Performance Modes of Electro-Acoustic Music	9
2.1.3 Discussion	15
2.2 Digital Audio Signal Processing	16
2.2.1 Digital Audio Signal Processing: General Overview	16
2.2.2 Hardware Based Digital Audio Synthesis	21
2.2.3 Software Based Digital Audio Synthesis	22
2.3 Control Environments for Real-Time Electro-Acoustic Music Systems	26
2.3.1 Historical Background	27
2.3.2 Music Instrument Digital Interface (MIDI)	28
2.3.3 MIDI Controllers	29
2.4 Programming Paradigms for Real-Time Electro-Acoustic Music Systems	31

2.4.1 Control Classes-----	31
2.4.2 Control Structures-----	33
2.4.3 Control Sources-----	33
2.5 Discussion-----	34
2.6 Max Signal Processing (Max/MSP) Programming Environment-----	36
2.6.1 Historical Background-----	37
2.6.2 MAX/MSP: Towards a Unified Environment and New Representation for Real-Time Digital Audio Signal Processing and Dynamic System Control-----	38
 3. Developing a Prototype Solution for Live Electro-Acoustic Music System in the MAX/MSP Programming Environment: A Case Study-----	 40
3.1 "Yonge Street Variations": System Analysis-----	41
3.2 General System Design-----	44
3.3 Digital Audio Signal Processing: Designing a Software Implementation of the External Hardware-----	47
3.3.1 Sampler Instrument Design-----	47
3.3.2 Audio Effect Processor Unit Design-----	50
3.3.3 CD-ROM Drive-----	51
3.3.4 Mixing Board-----	52
 3.4 Dynamic System Control: Developing Control Structures-----	 54
3.4.1 Control of Time-Based Events-----	54
3.4.2 Control of MIDI based events-----	55
3.4.3 Control of DASP events-----	57
 3.5 Graphical User Interface-----	 59

3.6 CPU Performance and System Optimization	62
3.7 System Evaluation	63
4. Conclusion	65
5. References	67
6. Bibliography	70
Appendix A “Yonge Street Variations”: System Setup	76
Appendix B “Yonge Street Variations”: Score	78
Appendix C “Yonge Street Variations”: Technical Requirements	96
Appendix D CD-ROM Content	97
Appendix E CD-ROM	98

1. Introduction

Since its inception, live electro-acoustic music has presented many hurdles to composers, performers and music teachers. The limitations of standard notation, the lack of a versatile delivery medium, gaps in performance tradition and music education, led to the situation where most electro-acoustic pieces had to be performed under the composer's guidance, and were often literally "unperformable" without the composer's assistance.

The rapid development of music and computer technologies, and its influence on industries, changed and improved electro-acoustic music hardware and software each year. At the same time, progress in the music and computer industry brought many hardware and software dependency problems in live electro-acoustic music performances since many original compositions required some original specific hardware and software which is no longer widely available. This situation led again to the case where the live electro-acoustic music performance process was not only guided or assisted by composers, but very often composers became the main suppliers of their own equipment for specific concerts or installations.

The revolutionary changes in music and computer technologies in the 1980's opened a new era for the development of live electro-acoustic music. The introduction and subsequent development of desktop computing, Musical Instrument Digital Interface (MIDI), digital audio, and Compact Disc (CD) technologies with all varieties of the format (CD-DA, CD-ROM, ECD and others) gave new possibilities for live electro-acoustic music. The increase of microprocessor power in the late 1980's and 1990's enabled real-time digital audio signal processing (DASP) and the development of new real-time digital audio signal processing programming environments. The release of MAX Signal Processing (MSP) for the Apple PowerPC platform in 1997 was a culmination of this process and combined a real-time desktop DASP and dynamic control in one programming environment.

Though some problems of live electro-acoustic music have been discussed in the past (Austin and Washka 1996 [1], Pennycook 1997 [2], Lippe, Settel, Puckette, Lindemann [3]) many of these issues remain relevant. To my knowledge, no fully satisfactory solutions to hardware/software dependence problems have been developed. The primary objective of this thesis is to undertake research and to examine new possibilities for hardware/software independency, and to develop a prototype solution using the real-time digital signal processing environment, MAX/MSP.

1.1 Thesis Structure

This thesis consists of a written document with an accompanying CD-ROM. The text provides information related to live electro-acoustic music, real-time digital audio signal processing, and a discussion of the software implementation of the “Yonge Street Variations” real-time electro-acoustic music system. The accompanying CD-ROM (Appendix E) contains the complete software implementation of the “Yonge Street Variations” real-time electro-acoustic music system and all the original software resources. For more detailed information see Appendix D.

2. Real-Time Electro-Acoustic Music Systems

2.1 Electro-Acoustic Music

In spite of the fact that electro-acoustic music has existed since the beginning of the century, numerous current sources, including some dictionaries dedicated to music technology and electro-acoustic music, contain different interpretations of the same term. For unknown reasons, some basic terminology and many important terms such as "electro-acoustic", "electronic" and "computer music" are even omitted. Although the goal of the given research does not include the detailed analysis, criticism and listing of all incomplete and misinterpreted sources, the decision has been made to provide some basic definitions related to this work to avoid any confusion. Since the definitions in many reference sources are attached to some specific periods in the history of electro-acoustic music (for example *musique concrète*), the proposed terminology will be given in the most abstract form and can be applied to any kind of music, independent of a specific historical period. Some of proposed definitions are based on "Introduction to Electro-Acoustic Music" by Barry Schrader [4], which provided the most adequate set of definitions at the time of its publication (1982). In order to maintain a more structured classification, according to the one specific feature, the following classification and some definitions will be proposed.

2.1.1. General Classification and Selected Definitions

Electro-Acoustic Music

Although there are a limited number of dictionaries related to music technology and electronic music [4] most of them, as well as most of the reference sources, omit this term. In the book “Introduction to Electro-Acoustic Music”, B. Schrader defines "electro-acoustic" as:

“any music that is produced, changed or reproduced by electronic means” [5].

This term is one of the most acceptable definitions of electro-acoustic music since it does not limit this genre of music to a specific sound generation source, and basically views electro-acoustic music as any kind of music, including acoustic, that is affected by electronic means. According to this definition, there are two meanings of electro-acoustic music:

- a) all the music produced by electronic music instruments, computers; as well as acoustic music, which is affected by electronic devices, such as special effects units (reverberation, delay, etc);
- b) all the music that is recorded, reproduced or transmitted via electronic equipment (tape recorder, CD player, radio, etc).

The first meaning of electro-acoustic music will be used in the given research.

According to the sound generation source, electro-acoustic music can be divided into the following subclasses: electronic music, musique concrète, and music for acoustic instruments and electronic source. The many “pure” examples of electro-acoustic

music subclasses which can be found in the period of the 1950's – 60's should be emphasized. Later, many compositions became more integrated or “mixed”, although some examples of “pure” electro-acoustic music subclasses can be seen at the present time.

Musique Concrète

The term of *musique concrète*

“refers to any electro-acoustic music that uses acoustic sounds as a source material” [5].

In other words, any music based on recorded acoustic sounds with sound manipulation by change of speed, direction, etc. can be classified as *musique concrète*. This term was proposed by Pierre Schaeffer in 1948 and in most reference sources, *musique concrète* refers to the French school of electro-acoustic music and is associated with this historical period. The proposed definition anticipates the use of this term in a more broad meaning.

Electronic Music

The definition of “electronic music” can be viewed as a music

“in which the source, or original, sound material has been electronically produced” [5].

In most cases the sound generation is performed by means of electronic oscillators that produce an electrical signal with subsequent transformation into a physical vibration. There are other terms for electronic music such as “electroponic music” used in the

United Kingdom and “tape music” which is more common in North America. Although last term is widely used in North America, it has several meanings and therefore will be discussed later. A lot of sources use the term electronic music with reference to the Cologne and German school of electro-acoustic music, however in this research the term will be used according to the definition given above.

According to the source of the generation device, there are two types of electronic music: synthesizer and computer music [5].

Synthesizer Music

Synthesizer music is a subtype of electronic music produced by a synthesizer, an electronic music instrument which is

“a system of electronic circuits, either analogue or digital, or a combination of both for the creation of sounds, using one or more methods of sound synthesis”[6].

The synthesizer consists of many oscillators which produce electronic signals as well as several other devices for altering and modifying the quality of the sound material.

Computer Music

By definition, “computer music” is a subtype of electro-acoustic music

“in which a computer is used to generate the sound material” [5].

Usually it is done by some special Digital Signal Processing (DSP) hardware, however some computer models do not require any additional hardware and have built-in sound

capabilities. Also, it should be mentioned that in many sources, the term “computer music” includes the music composed for instruments with aid of a computer. However, to avoid any confusion, this term will be used strictly in relation to digital audio signal processing.

Tape Music

Very often, many of the electro-acoustic compositions used some acoustic and electronically produced sound material, and therefore in 1952 a new term, “tape music”, was created by Vladimir Ussachevsky. The term “tape music” refers to a composition

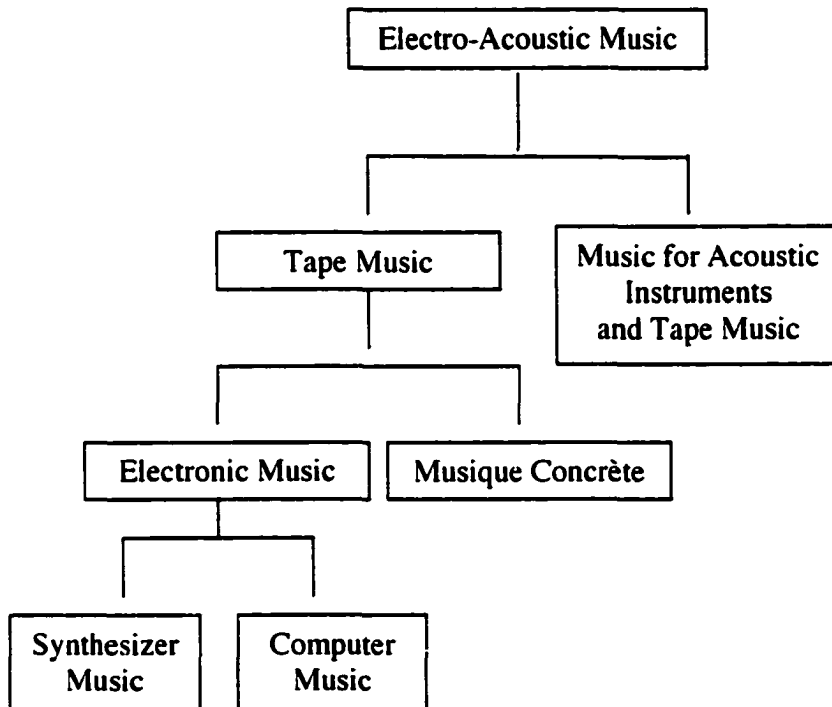
“that uses recorded sound material whether it is acoustic or electronic in origin.” [5].

The origin of this definition goes back to the 1950’s and is based on the meaning “composition for tape recorders”, confirming that this kind of music is dependent on tape recorders. The compositional process is done in the studio and the “score” is stored on the tape and can be reproduced by the means of tape recorder only (at the present time different formats of digital audio (DAT, ADAT, CD and hard drives are used).

Music for Acoustic Instruments and Tape Music

Since many electro-acoustic compositions use some acoustic instruments, the other type of electro-acoustic music is defined as a music for acoustic instruments and tape music.

According to the generation sound source the following classification of electro-acoustic music is proposed:



2.1.2 Performance Modes of Electro-Acoustic Music

Depending on the performance mode electro-acoustic music can be divided into two main categories: live (or real-time) and tape (or non-real-time) electro-acoustic music.

Tape or Non-Real-Time Electro-Acoustic Music

Tape or non-real-time electro-acoustic music is a performance mode of electro-acoustic music in which the actual performance is done in the form of simple playback, usually by means of some recording/reproduction device (tape or DAT recorder, CD player, etc.), and some medium carrier (any kind of analog or digital tape, CD, etc.). In order to bring some “live” aspect to the tape music performance, sometimes it can be mixed “in live”, using “diffuison” technique. Tape music composition was very common in the early days of electro-acoustic music when, due to limiting technology factors, real-time performances were not possible and computer resources were not powerful enough to handle real-time signal processing. However at the present time there are many composers working in the domain of tape music.

Live or Real-Time Electro-Acoustic Music

A performance mode of electro-acoustic music that requires any real-time/live performance or interaction with a human factor can be viewed as live or real-time electro-acoustic music. Depending on the complexity and form of the electro-acoustic music system, its performance and interactivity factor, real-time electro-acoustic music can be divided into the three following categories:

- a) tape music with acoustic instruments;
- b) real-time electro-acoustic music (with or without acoustic instruments);
- c) interactive electro-acoustic music (with or without acoustic instruments).

Before the defining all these types of real-time electro-acoustic music, the terms should be discussed.

Real-Time and Interactive

Although from one point of view these definitions are quite simple and obvious, there is often some confusion about the relationship of these terms. Most the music technology dictionaries [4] and other sources do not provide precise explanations. The term "real-time" refers to an action where no time delay is perceived by human factor. The interactivity state requires two independent sources, where output of one of them is dependent on the input of another. The real-time state is a premise for the interactivity state.

Tape Music with Acoustic Instruments

This is the simplest form of real-time electro-acoustic music, where the real-time factor is brought by performance on acoustic instruments only and the tape music is performed using traditional playback techniques.

Real-Time Electro-Acoustic Music (with or without Acoustic Instruments)

The real-time mode of this type of electro-acoustic music performance relates to the real-time electro-acoustic music system, which is a system controller or sound source generator of electro-acoustic music composition. The other form of this type can be real-time performance on electro-acoustic instruments (synthesizers, samplers, etc.). The performance can be with or without acoustic instruments.

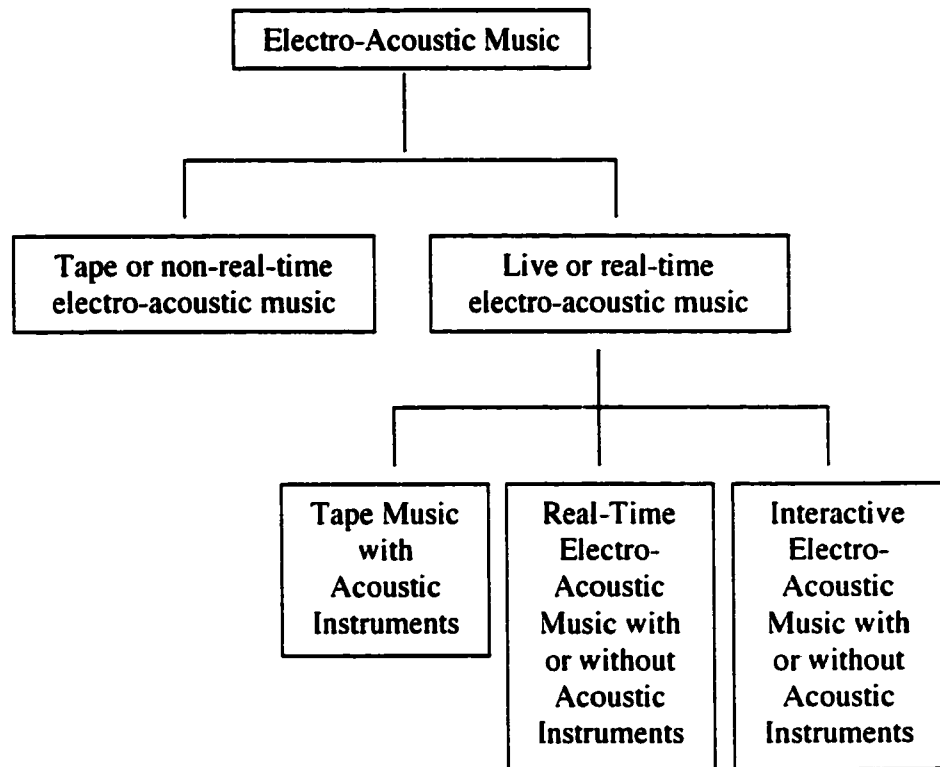
Interactive Electro-Acoustic Music (with or without Acoustic Instruments)

This is the most complex and advanced type of interactive electro-acoustic music. It refers to the real-time music performance with the use of interactive computer music systems. According to Rowe,

“Interactive computer music systems are those whose behavior changes in response to musical input” [7].

Depending on the given input the system will produce a certain output and this is the main driving force of interactive electro-acoustic music composition.

The general classification of electro-acoustic music, according to the performance mode is illustrated in the following chart:



Electro-acoustic Music and Multichannel Audio

Depending on a number of output channels, electro-acoustic music can be one channel (mono) and multichannel (often this term refers to more than two channels). For the different number of channels some specific terminology is used. For instance: stereo (two channels), quadraphonic (four channels), octaphonic (eight channels) and ambisonic (two- or three-dimensional, varied number of channels and loudspeakers from 4 upwards) [8].

Presentation Forms of Electro-Acoustic Music and Other Forms of Electronic Arts

According to the form of presentation there two genres of electro-acoustic music: performance (stage presentation) and installation.

Electro-Acoustic Music Performance

Electro-Acoustic Music performance presentation is a traditional form of a concert performance, where the piece is performed in some specific place during a certain period of time.

Electro-Acoustic Music Installation

Installation is a form of electronic arts that involves some specific setup and “tuning” of a certain equipment which reacts accordingly to the given input. Usually there is no specific timeframe for the installations. The quality of installation output is dependent on the familiarity and training of the user, who provides input to the system.

The provision of a set of definitions related to electro-acoustic music in other fields of electronic arts where the sound is an integral and important part of the overall piece should be mentioned.

Audiovisual (Multimedia) Genre

During recent decades there has been a rising interest in different means of expression which employ sound, image, and text. Recent technological developments have

successfully enabled a new type of electronic art combining simultaneously several different media. Depending on the style and goal of the work, audio can be an important part of multimedia composition and therefore all the above aspects of electro-acoustic music can be applied to the audiovisual arts.

2.1.3 Discussion

Since its inception the development of electro-acoustic music was not paralleled by the development of appropriate theoretical and musicological research. Due to their conservative and traditional backgrounds based on the music history and theory of classical European art music, most musicologists and music theorists have tried to avoid topics related to the field of electro-acoustic music. Many music schools do not provide an appropriate degree in this field and therefore a theoretical and musicological branch of electro-acoustic music in the traditional meaning has not developed. Most of the work in the history and theory of electro-acoustic music has been written by electro-acoustic music composers and software developers, and can therefore be very subjective without a rigorous methodology. Hence many uncharted territories still exist, including a solid classification of the areas making up electro-acoustic music, clearly stated definitions of basic terminology, and many other topics related to the form of electro-acoustic music composition, aesthetics, etc. Many of these issues could be potential topics for further musicological research.

2.2 Digital Audio Signal Processing

Digital audio signal processing belongs to the fields of science and engineering. During the last two decades, rapid developments within the computer industry have brought to the end user more powerful, faster, smaller and cheaper digital hardware and software which enable wide applications of computer technologies, including the music field. Therefore, at the present time, inexpensive, portable and relatively fast desktop-based digital audio signal processing is becoming more popular and accessible in the professional and amateur music world.

2.2.1 Digital Audio Signal Processing: General Overview

Similarly to the situation in electro-acoustic music terminology and classification, in many primary sources and dictionaries related to digital audio signal processing clear definitions, well-organized classifications and some important terms are missing. In many sources [9] the term “digital audio signal processing” does not appear and in most cases it is associated with the definition of digital signal processing, which has a broader meaning, including such categories as speech processing, signal communications, etc. Thus, according to Freedman, digital signal processing is

“a category of techniques that analyze signals from sources such as sound, weather satellites and earthquake monitors. Signals are converted into digital data and analyzed using various algorithms such as Fast Fourier Transform.”
[10].

Digital Audio Signal Processing (DASP)

Since in numerous sources an adequate definition of DASP is omitted, a possible definition might be the following:

Digital audio signal processing is as a process that includes digital sound synthesis or sampling, as well as the reproduction, transformation or transmission of acoustic signals by means of digital processing devices such as computers, effect processors, synthesizers, samplers, etc.

Depending on processing time, digital audio signal processing can be either real-time or non-real-time. The real-time mode requires intensive computational power and became available in the simplest form of digital sound generating devices beginning in the 1970's. The more integrated and complicated forms of real-time digital audio signal processing became available in the 1980's with extensive development occurring in the 1990s.

The basic operations of most signal processing techniques, whether analog or digital, consist of simple mathematical operations such as multiplication and time delay. In the analog domain these operations are achieved through the use of amplifiers and reactive circuit elements. In the digital domain multiplication is accomplished using hardware multipliers, while delay requires storing a digital number for some value of sample periods.

In general, digital audio signal processing as applied to music may be divided in the following categories:

1. sound synthesis;
2. sampling;
3. effects processing;
4. signal spectral analysis;
5. digital audio recording and reproduction;
6. digital audio signal transmission;
7. digital audio coding formats.

Sound Synthesis

According to Dodge, sound synthesis is the process of

“realization of electronically generated acoustical material” [11].

Sound synthesis may be analog, or digital, and the sound may be implemented in either hardware or software. Concerning the time required for signal processing, sound synthesis may take place in real-time or non-real-time. In general, sound synthesis techniques can be classified into the following categories:

1. frequency modulation synthesis;
2. amplitude modulation synthesis;
3. additive synthesis;
4. subtractive synthesis;
5. synthesis by waveshaping;
6. cross synthesis;

7. granular synthesis;
8. physical modeling synthesis;
9. formant synthesis.

Since sound synthesis is not the main subject of this research, none of the above categories will be discussed, however some aspects of sound synthesis that have more pertinence to the history and development of electro-acoustic music systems will be reviewed in the next sections.

Sampling

Dodge defines sampling as

“ a process of representing a waveform by measuring its value at discrete points in time” [11].

In other words sampling is a process of digital recording and reproduction of an external sound with a possible frequency modification.

Effects processing

Effects processing is a technique based on altering and signal processing of already existing sounds. In most cases, effects processing occurs in two dimensions: time (reverberation, delay, etc) and frequency (flanging, chousing, etc).

Signal spectral analysis

Signal spectral analysis refers to the characterization of the frequency content of a signal. It has wide ranging applications from electro-acoustic music to radar signal processing, etc. One of the most common methods of spectral analysis in music is the Fast Fourier Transform.

Digital audio recording and reproduction

Digital audio recording and reproduction became very popular with the introduction of digital audio tape recorders and compact disc players. One of the main advantages of this aspect of DASP is preservation of the original quality of recorded material during all stages of audio post-production.

Digital audio signal transmission

Digital audio signal transmission relates to the field of digital communications and includes such subjects as digital audio broadcast, telecommunications, etc.

Digital audio coding formats

Digital audio coding formats are different types of compression techniques that are used for audio storage, reproduction and transmission. Some of the most common coding formats for digital audio are MPEG type formats, AC-3, Real Audio, etc.

Since most electro-acoustic music is based on sound synthesis techniques and signal effects processing, the next sections will be devoted to hardware and software based sound synthesis and its historical development.

2.2.2 Hardware Based Digital Audio Synthesis

Digital sound synthesis was introduced in 1957 at Bell Laboratories. Due to the limited computational power at that time, to produce even a short sound was a very lengthy and complex procedure. Starting from the 1970's, new developments in computer technology enabled real-time digital audio signal processing. In 1971, the Allen Computer Organ was produced by the Allen Organ Company and North American Rockwell. This was the first instrument based on digital table-lookup oscillators with programmable waveforms. During the following years, intensive development in the field of digital sound led to the development of new digital synthesizers in some research facilities: Dartmouth College (1973), Massachusetts Institute of Technology (1974), Institute de Recherche et Coordination Acoustique/Musique (1976), Center for Computer Research in Music and Acoustics at Stanford University (1977) among others. In most cases, these new digital synthesizers were very large and quite expensive. At the same time, commercial synthesizers were becoming available, including the Synclavier (New England Digital Corporation, 1977), and the GS1 (Yamaha, 1980). In the late 1970's the low cost of semiconductor memory enabled the introduction of first digital sampling instruments: the Synclavier II (1979), the Fairlight (Fairlight Computer Music Instrument, 1979). Their price remained quite high, however. Another commercial instrument which should be

mentioned is the keyboard synthesizer DX7 (Yamaha, 1983), based on the frequency modulation technique, which was extremely successful and relatively inexpensive.

Viewing the historical development of digital synthesis hardware it can be observed that in the 1980's there were two groups of digital synthesizers:

- a) those designed on a single synthesis technique and which are relatively inexpensive;
- b) those based on a sample playback principles and which are quite expensive due certain technological limitations.

The following development of music technology in the 1990's resolved many problems and both types of digital synthesizers became available for general use.

2.2.3 Software Based Digital Audio Synthesis

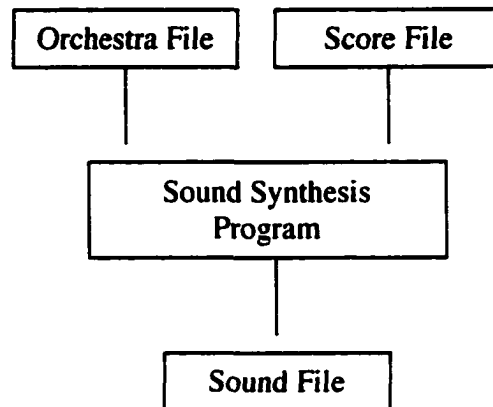
In general, software based digital audio synthesis programming environments can be grouped in two main categories:

- a) synthesis programming languages;
- b) software synthesizers.

Another type of software synthesis programming language – control sound synthesis programming environment - will be discussed in the section “Control Environments for Real-Time Digital Audio Signal Processing”.

The origin of synthesis programming languages lies in the 1960's and revolves around Max V. Mathews and his team at Bell Telephone Laboratories in New Jersey, where in 1960 the original generator language, Music III, was developed. Since then many sound synthesis languages have been developed based on Music III, and are well known as the Music N family. Among the most popular of these languages are Music IV (Mathew, Miller, 1963), Music V (Mathew, Miller, 1966), MUS 10 (Chowning, Poole, Smith, 1966), Music 4BF (Howe, Winham, 1967), Music 11 (Vercoe, Haflich, Hale, Howe, 1973), Cmusic (Moore, Loy, 1980), MIX (Lansky, 1982), Cmix (Lansky, 1984), Csound (Vercoe, Karstens, 1986), and Common Lisp Music (Schottstaedt, 1991), SuperCollider (McCartney, 1996). Although each new version of Music N added some new features and refinements, the basic aspects of the general structure remained the same.

The programming language of the Music N family is normally divided in two parts: the score language and the orchestra language. The score language defines the note list (instrument names, frequency list, start and duration times and other parameters of events associated with time and pitch). The orchestra language specifies instrument design and relates to the specific sonic structure of defined instruments. The basic functionality of the Music N programming language is illustrated in the following diagram:



Although the listed sound synthesis languages are non-real-time, they became quite popular and some of them (e.g. Csound, Cmix) are still currently in use. One of the most valuable features of the Music N family is its cross-platform nature. For instance the same Csound file can be processed on Macintosh, Unix or Windows platforms.

The second category of software, based sound synthesis is the software synthesizer, which began to appear in the second half of 1990's when the increasing speed of microcomputers enabled real-time sound synthesis. This group of software can be divided in two subcategories:

- a) simple playback software synthesizer;
- b) more advanced software synthesizer.

The first subcategory includes software playback synthesizers developed as cheap hardware replacements and is used primarily for the simple playback of MIDI files. Some examples of this group are Apple QuickTime Software Synthesizer, the Roland

Virtual Sound Canvas VSC-550, and the Yamaha S-YXG. In comparison with real software synthesis programming environments, in most cases the functionality of these low cost solutions is quite limited.

The second subcategory includes more advanced software synthesizer software, such as ReBirth RB-338 by Steinberg or MetaSynth 2.5 by Arboretum Systems. These software applications are more expensive and have more advanced features. Their graphical user interfaces provide simple access to their operational functionality and enables faster results with less time and programming effort.

2.3 Control Environments for Real-Time Electro-Acoustic Music Systems

In addition to real-time digital audio signal processing, system control and time scheduling are two of the most important aspects of live electro-acoustic music systems. According to Roads [12], all the control programming environments for real-time DASP can be grouped into the two following categories, according to the architecture and design of the digital audio processing hardware:

- a) programming languages for hardware with fixed function DASP;
- b) programming languages for hardware with variable function DASP.

Fixed-function DASP hardware consists of a number of specific components (oscillators, envelope generators, filters, etc.) that may be interconnected in various configurations in order to process sound. Most of the synthesizers available on the market have fixed-function DASP hardware because of its inexpensive design. Programming software of this type is usually quite simple and is not as complex as software for variable-function DASP hardware.

Variable-function DASP hardware consists of a collection of functional units that may be interconnected or “patched” like the different modules of an analog synthesizer. This flexibility adds much more control over digital audio signal processing, but also results in more complex software. Although this type of hardware is more expensive, one of its main advantages is its flexibility, which allows for the possibility of future reprogramming if necessary. One of the most popular examples of this type of

hardware is the IRCAM Signal Processing Workstation (ISPW), which is still used by many electro-acoustic composers.

A third category of hardware architecture for real-time digital audio processing should also be mentioned: a standard general purpose microprocessor without any special DASP card. In its basic programming, this type of hardware architecture is similar to the second type of control environment mentioned above, since the program can be changed if necessary. This type of architecture is becoming more popular since it does not require any additional hardware.

Since the present text is concerned with the low-level control of real-time digital audio signal processing by resources based on the MIDI communication protocol, the historical background of this topic will be briefly sketched.

2.3.1 Historical Background

Although the first synthesizers began to appear in the 1960's, for approximately twenty years, a means of interconnection and information exchange between electronic music instruments was missing. The first examples of custom-built interfaces with appropriate software protocols can be found in hybrid synthesizers such as the GROOVE system (Mathews and Moore, 1970) and the HYBRID system (Kobrin, 1977), however these are among very few examples. In the late 1970's the issue of interconnection of electronic music instruments began to become more critical. Music

software developed for one system could not be used by other music systems and no standard way existed for the real-time performance synchronization of electronic music instruments. The establishment of an electronic music instrument communication protocol became a necessity.

In the early 1980's a few major electronic instrument companies such as Sequential Circuits, Oberheim, and Roland Corporation, decided to develop a joint communication protocol, which led to the establishment of Musical Instrument Digital Interface (MIDI) in 1983.

2.3.2 Musical Instrument Digital Interface (MIDI)

Musical Instrument Digital Interface (MIDI) [13] is a communication protocol for music representation in symbolic form. It can be described as an interconnection scheme between electronic music instruments and computers for data exchange and as a language for transmitting musical structures between MIDI devices.

The Musical Instrument Digital Interface consists of two following components:

- a) communication protocol (MIDI language);
- b) hardware interface (MIDI connectors).

The MIDI communication protocol is a language for the description of musical structures in a binary form. Each word describing an element of a musical performance such as pitch, duration, velocity, etc., is translated into a specific binary

code. MIDI messages may also include information selecting different instruments to play, and to control various performance parameters of electronic musical instruments.

The hardware interface of the MIDI communication protocol stipulates the use of 5-pin DIN connectors in a specific configuration, according to the MIDI 1.0 Specification.

There is a third component of the MIDI communication protocol: a distribution format, called the Standard MIDI File Format (SMF). Although the Standard MIDI File Format is slightly different from native MIDI protocol, this format allows users to store MIDI messages from one or many MIDI devices whether in real-time or non-real-time mode, with the possibility of future reproduction.

The establishment of the MIDI communication protocol is an important achievement for the music industry and enables the real-time control of MIDI electronic music instruments.

2.3.3 MIDI Controllers

There are two possible definitions of a MIDI controller:

- a) an electronic music instrument, usually without built-in sounds, designed for the MIDI control of devices such as MIDI modules and samplers, among others.

- b) a type of MIDI event: control-change messages that change different aspects of the sound such as volume, vibrato, etc.

The first definition includes different types of music instrument controllers such as keyboard-controllers, guitar-controllers, wind-controllers, string instrument controllers, and drum-controllers. Depending on the type of instrument, controllers can control different parameters of a sound and have different functionality. For instance, some features existing in a wind-controller might be absent in a keyboard-controller and vice versa. For the control of high-level MIDI events other controller types also exist, such as audio-signal controllers (pitch converters), computer periphery input devices (keyboard, mouse, etc.), and many other different sensor-based controllers (for motion, pressure, temperature, etc.).

The second meaning of the MIDI controller refers to the relatively low-level control (without accessing DASP state) of MIDI control-change messages.

In general there are two types of control information: discrete and continuous. While discrete information is based on discrete “on” and “off” actions such as pressing a key or foot-switch pedal, continuous control information refers to gradually varying actions which are made by sliders, wheels, faders and pedals.

2.4 Programming Paradigms for Real-Time Electro-Acoustic Music Systems

According to their user interface, all programming environments for real-time electro-acoustic music systems can be classified into one of three main categories:

- a) data based;
- b) text based;
- c) icon based.

The first category includes playback software that provides a restricted level of control over real-time performance. Examples of this type include sequencing or audio playback software such as Opcode Studio Vision, Bias Peak, etc.

The second category includes text-based programming environments with programming languages that use a written syntax. In order to write code the end user has to preserve all the rules and structures of the particular language. Text-based programming environments are usually based on a programming language like C, Lisp, etc. Most of the languages developed in the late 1970's and early 1980's are text based, for instance Music 1000 (Wallraff, 1980) and Music 400 (Puckette, 1982).

The last category, icon based programming languages, started to appear as microprocessors began to increase in speed. In general they have the same functionality as text based languages, however a graphical user interface, a set of graphical objects, enables more intuitive and flexible programming. All the

programming environments of the Max family (IRCAM Max, Max, MSP) have icon-based graphical user interfaces. Another important programming language of this category is Kyma, which was developed by Carla Scaletti.

2.4.1 Control Classes

There are two main control classes of programming paradigms for real-time electro-acoustic music that can be described as having low-level and high-level system control.

The high-level control class is similar to the definition of the control programming languages for hardware architecture with fixed function DASP that was explained in the previous section. In the most cases it is event-based control, such as MIDI and possibly other protocol messages, depending on system design. This type of control does not include low-level control of DASP state.

Low-level control classes allow the alteration and modification of digital audio signal processing in real-time. Like programming environments based on hardware architecture with variable function, this class gives complete control over the most low-level functions of real-time DASP.

2.4.2 Control Structures

According to their control structure design, programming paradigms for live electro-acoustic music systems can be grouped into two main categories:

- a) those based on direct control structures;
- b) those based on conditional control structures.

The first category implies the presence of a direct input signal from any possible source (audio, MIDI, system event, etc.)

The other category, based on conditional control structures, consists of two subcategories: time-based and event-based. The time-based subcategory generates control structures according to a time line whose values may be global or local. The event-based subcategory generates control structures which react to a certain event type, such as a frequency, duration, dynamic or any combination of these types.

2.4.3 Control Sources

Control source paradigms of real-time electro-acoustic music systems can be divided in two groups:

- a) those based on live input;
- b) those based on generative events.

The first group of control sources (live input based) utilizes live input such as audio signals, MIDI messages or any other system events including pressing keys, clicking a mouse, etc.

The second group (generative based) includes programming paradigms for algorithmic composition that are integrated into a real-time DASP control system.

2.5 Discussion

As part of a critical evaluation of the development of digital audio signal processing and real-time system control, the following conclusions may be drawn:

- a) as a result of prohibitive cost and lack of portability, for many years the domain of digital audio signal processing was restricted to a limited number of electronic music studios and research centers;
- b) with the development of the computer industry at the end of 1980's, issues of portability gradually became less significant, however the high cost of hardware still remained a deterrent;
- c) the introduction of the MIDI protocol in 1983 was a major event in the development of the music industry;
- d) the rapid development of computer technology starting in the middle 1980's enabled desktop-based DASP, however the need for unified and integrated environments for DASP and dynamic control still remained.

Currently advanced computational power and portability of personal computers combining both MIDI and real-time DASP present a new landscape for electro-acoustic music.

2.6 MAX Signal Processing (MAX/MSP) Programming Environment

“Max is a graphical music programming environment for people who have hit the limits of the usual sequencer and voicing programs for MIDI equipment.”

— Miller Puckette, Max reference manual, 1988 [13]

Max is an object-oriented programming environment for MIDI based system control applications developed for the Macintosh computer platform. The Max programming language is named after Max V. Mathews, who has made important contributions to the field of digital audio signal processing and real-time control music systems. The language has an icon-based user interface, provides real-time control and has a wide range of applications from controlling external devices such as laser disc players, synthesizers and samplers, to prototyping applications or signal data analysis.

According to the “Cycling’74” web site, the Max Signal Processing (MSP)

“is a set of extensions to Opcode's Max 3.5 environment that let you do real-time synthesis and signal processing with your PowerPC Mac OS computer.”

[14]

The combination of digital audio signal processing objects with the Max programming environment gives a great deal of dynamic control over digital audio signal processing

and makes this environment unique and extremely powerful for real-time electro-acoustic music systems.

2.6.1 Historical Background

The development of the Max programming environment for the Macintosh was begun by Miller Puckette at the Institut de recherche et de coordination acoustique/musique (IRCAM) in Paris in 1986. The original idea was to develop a non-graphical language to control IRCAM's 4X synthesizer. Later implementations, however, led to the development of a graphical environment for controlling MIDI on the Next and later the Macintosh computer platforms. In 1989, IRCAM began development of a real-time synthesizer card for the Next computer called the IRCAM Signal Processing Workstation (ISPW). Around this time Max was ported to the Next computer platform and ISPW, and a special set of digital audio signal processing objects was created. The Max programming language in combination with these audio extensions was known as Max/FTS and was widely used at IRCAM and in other music research centers and electro-acoustic music studios around the world. The current Macintosh version of Max was developed by David Zicarelli and was released by Opcode in 1991.

In 1996 Miller Puckette, working at the University of California San Diego, started development of the Pd (Pure Data) programming language that also supported real-time digital audio signal processing. Initially developed for the SGI platform, Pd was ported to the Windows NT platform and shortly after David Zicarelli decided to add a similar environment for the Apple PowerPC, expanding Opcode's Max language. As a

result, in 1997 the Max Signal Processing (MSP) environment was developed. Its infrastructure is similar to that of the Pd signal processing environment, and adds other features based on the Max/FTS programming language. In addition, MSP includes further innovations and graphical user-interface enhancements appropriate to the Opcode Max environment.

2.6.2 MAX/MSP: Towards a Unified Environment and New Representation for Real-Time Digital Audio Signal Processing and Dynamic System Control

The Max/MSP programming environment provides a high level graphical programming language with an icon-based user interface. The graphical user interface provides an intuitive programming environment in which programs may be written by simply connecting or “wiring” objects to each other. At the same time, it transforms the developed algorithms into efficient DASP code and enables real-time performance of complex electro-acoustic music structures.

The MSP extensions set includes over 75 objects that synthesize, process and analyze audio signals in real-time. The unique integration of the Max event scheduler and MSP digital audio signal processing provides an excellent unified environment for real-time DASP and a new level of dynamic control. The Max/MSP programming language enables high-quality program development with high-speed access to low-level DASP in real-time. The complete support of MIDI and other communication protocols

provides the ability to develop high-standard programming schedulers and to use different MIDI devices and various controller types.

Because it is based on the C programming language, Max/MSP allows additional objects to be developed by end users according to their individual needs.

Due to its power and flexibility, the Max/MSP programming has a wide range of potential applications: software and instrument design, education and research, sound design and composition. Max/MSP has thus become one of the most important environments for electro-acoustic music composers who need complete control over sound event structures.

3. Developing a Prototype Solution for Live Electro-Acoustic Music System in the MAX/MSP Programming Environment: A Case Study

In order to examine the Max/MSP programming environment and its potential for live electro-acoustic music, the decision was made to develop a prototype software solution for a specific live electro-acoustic music system that involves real-time digital audio signal processing and dynamic control. After reviewing several potential examples, the composition “Yonge Street Variations” by Bruce Pennycook was chosen as a complex, interesting example that requires in its original form many external audio and MIDI hardware devices. The main goal of the prototype solution was to solve such possible performance problems as hardware dependence and portability.

3.1 “Yonge Street Variations”: System Analysis

The composition “Yonge Street Variations” by Bruce Pennycook is written for amplified cello, electronics and real-time computer system and is the eighth in the *Praescio* series of pieces for one or more instruments and computer system.

A detailed analysis of the “Yonge Street Variations” was carried out using the following resources:

- a) performance setup documentation (see Appendix A);
- b) score (see Appendix B);
- c) concert recording.

According to this analysis of the composition and the required real-time music system, all the system components and programming structures can be seen as taking place in two dimensions: horizontal and vertical.

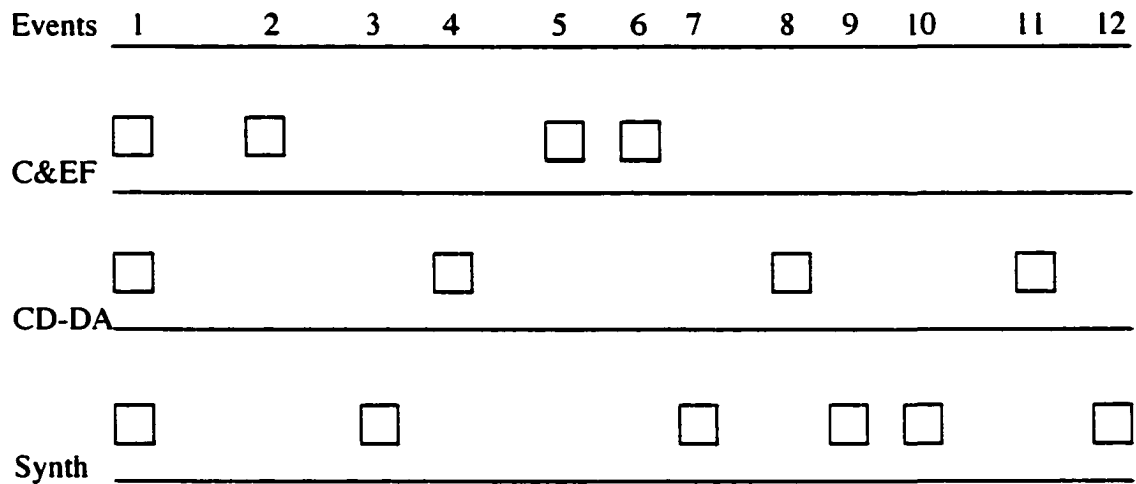
The horizontal dimension includes all the audio processing work and uses the following resources: one microphone, one synthesizer Roland JV-1080, one CD player, two effects processors (Lexicon LXP-15 and Digitech TSR-24) and a mixer (9 lines in, 2 sends and stereo out). For a more detailed representation see Appendix A.

The vertical dimension includes event structures and dynamic control using a foot-switch pedal, a MIDI interface, an Apple PowerPC computer and the performance software.

Following analysis of the performance setup documentation, the following three levels of audio signal processing were determined:

- a) Cello In & Effects Processing (C & EP),
- b) Compact Disc Digital Audio (CD-DA);
- c) MIDI Synthesizer (Synth).

Based on the score analysis and the concert recording, the compositional time line was divided into twelve major events and the following graphical representation of the music structures was made:



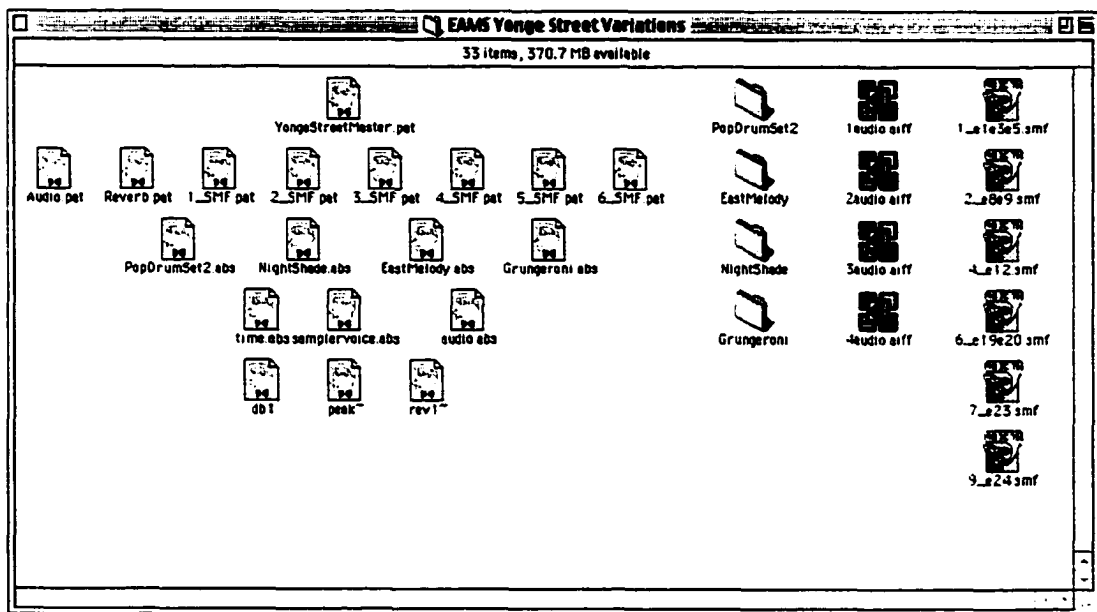
In order to make the “Yonge Street Variations” electro-acoustic music system portable, and to resolve the issue of hardware dependence, the decision was made to implement a “software version” of the system with complete integration of digital

audio signal processing and dynamic control. This version is discussed in the following sections.

3.2 General System Design

The data structures of the “Yonge Street Variations” real-time electro-acoustic music system can be grouped into the two following categories:

- a) programming structures (Max/MSP);
- b) data carriers (SMF, AIFF files).

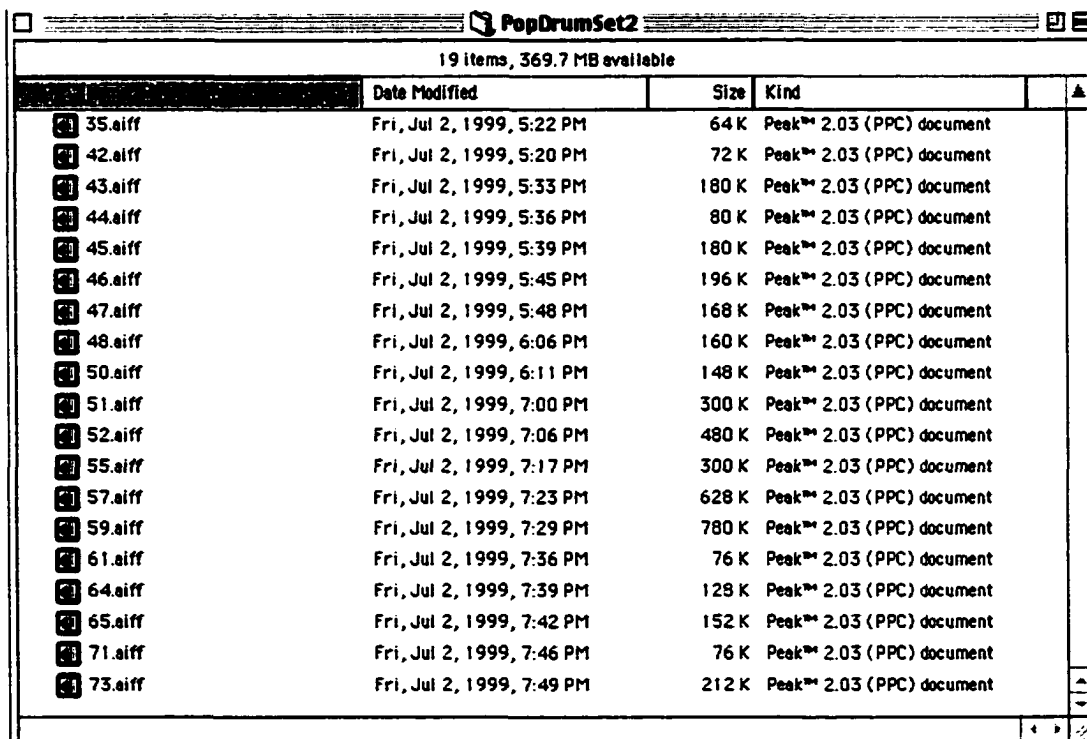


The first type of programming structure (Max/MSP) contains three subtypes, providing different levels of control and information access.

- a) high-level structures based on “.pat” files which provide a certain level of control according to the structural design (e.g “Yonge Street Master.pat”, “Reverb.pat”, “1_SMF.pat”;

- b) middle-level structures based on “.abs” files, which are the building blocks of the high-level structures (e.g. “PopDrumSet2.abs”, “samplervoice.abs”);
- c) low-level structures – small abstractions such as “dbl” and “revl~”, which perform a small specific task.

The second type – data carriers – are preprocessed files in two formats (SMF and AIFF) that are driven by the programming structures. Included in this type are all the instrument samples (AIFF files), which are placed in four folders according to instrument name (e.g. PopDrumSet2, EastMelody, NightShade, Grungeroni).



	Date Modified	Size	Kind
35.aiff	Fri, Jul 2, 1999, 5:22 PM	64 K	Peak™ 2.03 (PPC) document
42.aiff	Fri, Jul 2, 1999, 5:20 PM	72 K	Peak™ 2.03 (PPC) document
43.aiff	Fri, Jul 2, 1999, 5:33 PM	180 K	Peak™ 2.03 (PPC) document
44.aiff	Fri, Jul 2, 1999, 5:36 PM	80 K	Peak™ 2.03 (PPC) document
45.aiff	Fri, Jul 2, 1999, 5:39 PM	180 K	Peak™ 2.03 (PPC) document
46.aiff	Fri, Jul 2, 1999, 5:45 PM	196 K	Peak™ 2.03 (PPC) document
47.aiff	Fri, Jul 2, 1999, 5:48 PM	168 K	Peak™ 2.03 (PPC) document
48.aiff	Fri, Jul 2, 1999, 6:06 PM	160 K	Peak™ 2.03 (PPC) document
50.aiff	Fri, Jul 2, 1999, 6:11 PM	148 K	Peak™ 2.03 (PPC) document
51.aiff	Fri, Jul 2, 1999, 7:00 PM	300 K	Peak™ 2.03 (PPC) document
52.aiff	Fri, Jul 2, 1999, 7:06 PM	480 K	Peak™ 2.03 (PPC) document
55.aiff	Fri, Jul 2, 1999, 7:17 PM	300 K	Peak™ 2.03 (PPC) document
57.aiff	Fri, Jul 2, 1999, 7:23 PM	628 K	Peak™ 2.03 (PPC) document
59.aiff	Fri, Jul 2, 1999, 7:29 PM	780 K	Peak™ 2.03 (PPC) document
61.aiff	Fri, Jul 2, 1999, 7:36 PM	76 K	Peak™ 2.03 (PPC) document
64.aiff	Fri, Jul 2, 1999, 7:39 PM	128 K	Peak™ 2.03 (PPC) document
65.aiff	Fri, Jul 2, 1999, 7:42 PM	152 K	Peak™ 2.03 (PPC) document
71.aiff	Fri, Jul 2, 1999, 7:46 PM	76 K	Peak™ 2.03 (PPC) document
73.aiff	Fri, Jul 2, 1999, 7:49 PM	212 K	Peak™ 2.03 (PPC) document

In general, the software design of the “Yonge Street Variations” electro-acoustic music system consists of the two following parts:

- a) software implementation of the external hardware;
- b) design of control structures.

Both types are implemented in the Max/MSP programming environment with a high level of integration and dynamic control.

3.3 Digital Audio Signal Processing: Designing a Software Implementation of the External Hardware

The performance of the “Yonge Street Variations” real-time electro-acoustic music system requires four types of external hardware devices:

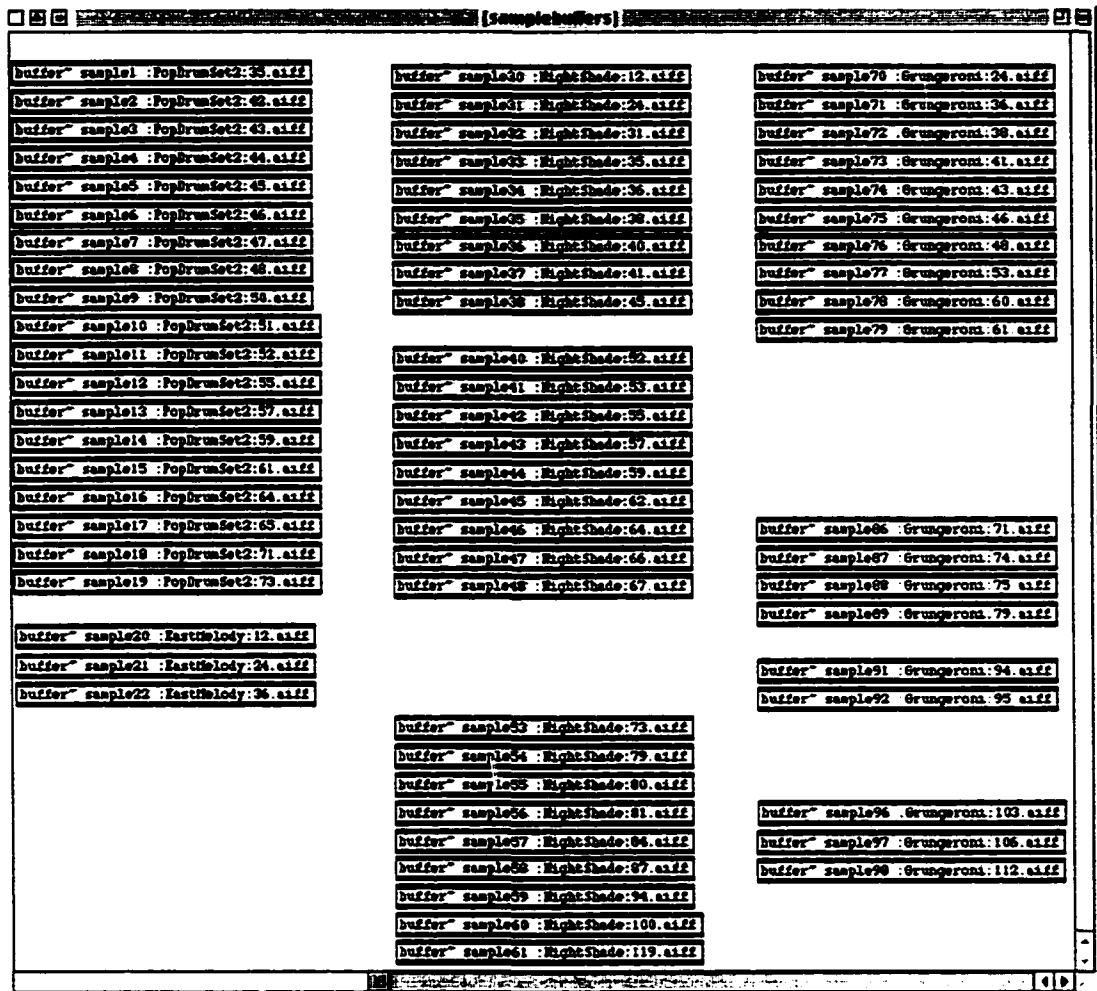
- a) MIDI synthesizer;
- b) audio effects processor;
- c) CD-ROM player;
- d) mixing board.

In order to solve the hardware dependence problems a software implementation of all these hardware types was done in the Max/MSP programming environment. Each type will be discussed separately.

3.3.1 Sampler Instrument Design

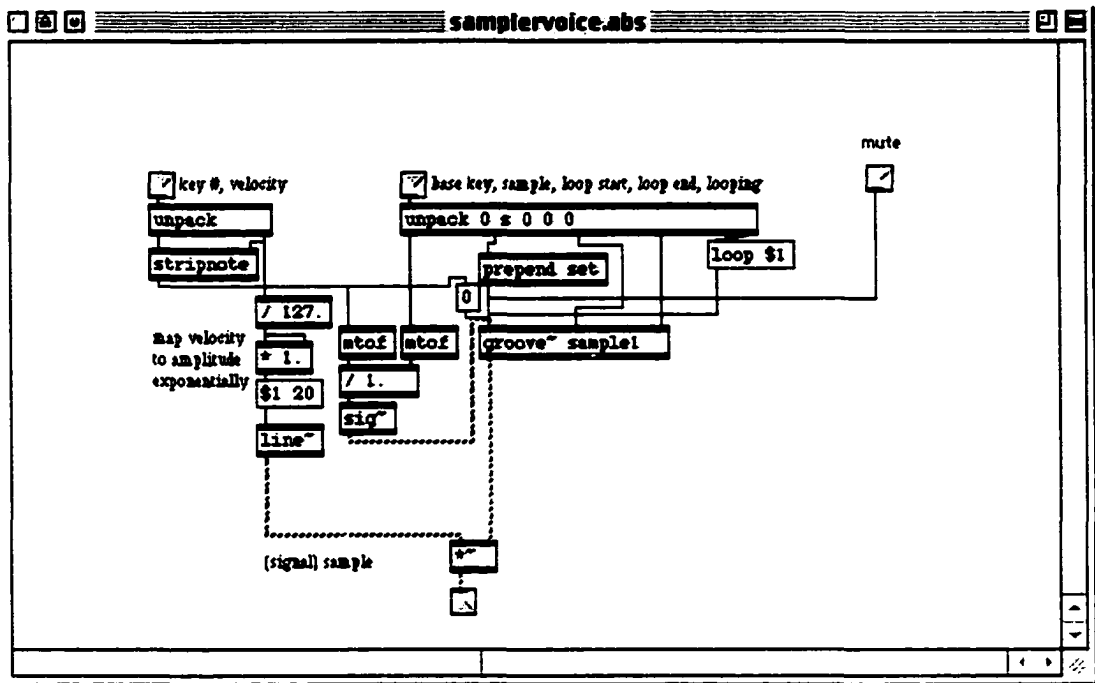
In order to recreate the synthesizer sounds as closely as possible, the decision to use a software sampler implementation was made. After a complete analysis of the six SMF files, all the note numbers played by each of the four instruments were collected and appropriate samples were made. All the sampling was done using the following equipment: Roland JV-1080 Synthesizer, Apogee PSX-100 A/D-D/A Conversion System and Tascam DA-30 MK II Digital Audio Tape Recorder with digital transfer via a MOTU 2408 audio interface to an Apple Power Macintosh G3 300MHz. In order to preserve the dynamic range of all the sounds the loudest note was recorded at the

highest level possible and all the samples were scaled accordingly. All the samples are preloaded into RAM to allow fast access.

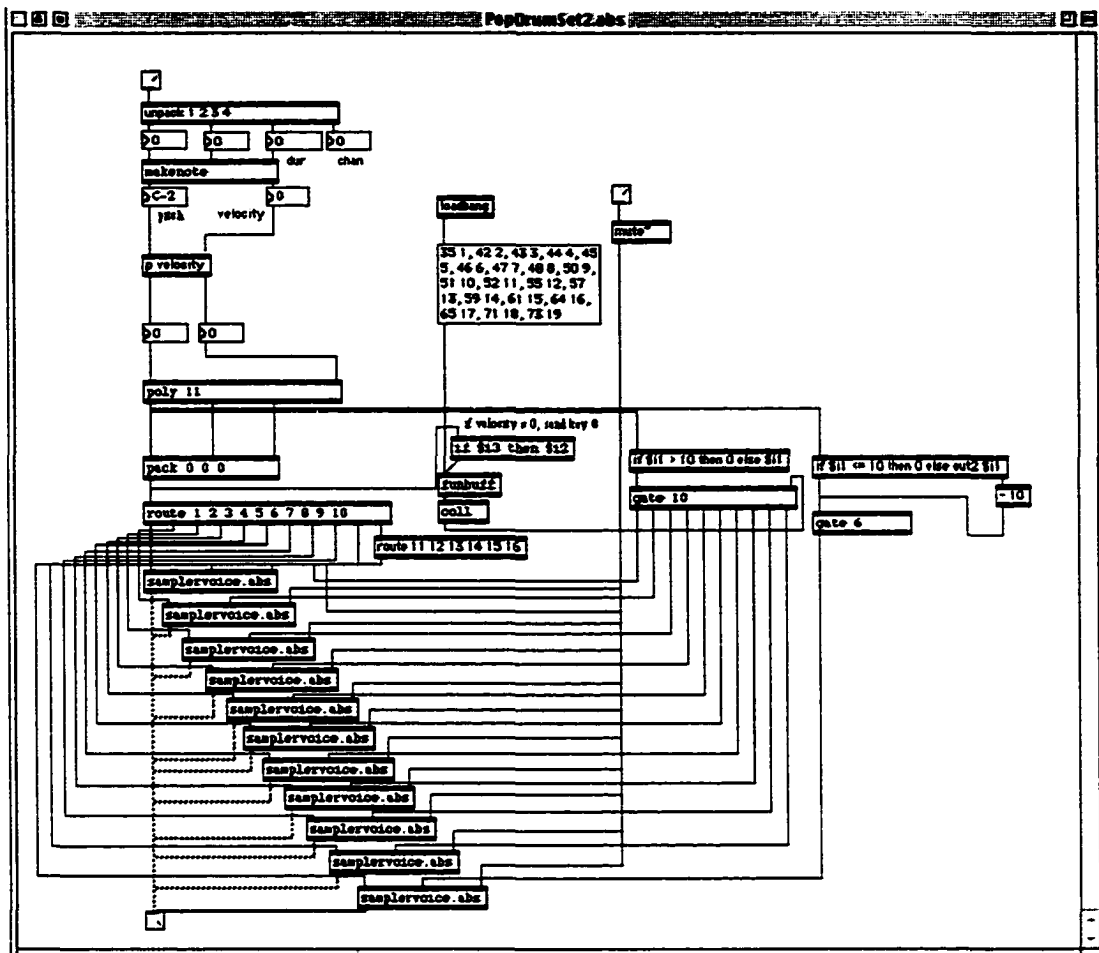


The implementation of the software sampler is based on the Tutorial 20 example provided with the MSP software [16]. The sampler design includes several levels of patches that perform tasks ranging from more general middle-level patches to lower-level patches that are specific to a certain instrument type.

The example of “`samlervoice.abs`” illustrates the first type and is used in all of the sampler instrument designs.



The second type is represented by the example "PopDrumSet2.abs". Depending on the instrument requirements, patches of this type may have different parameters, such as voice numbers, MIDI notes, special processing, etc.

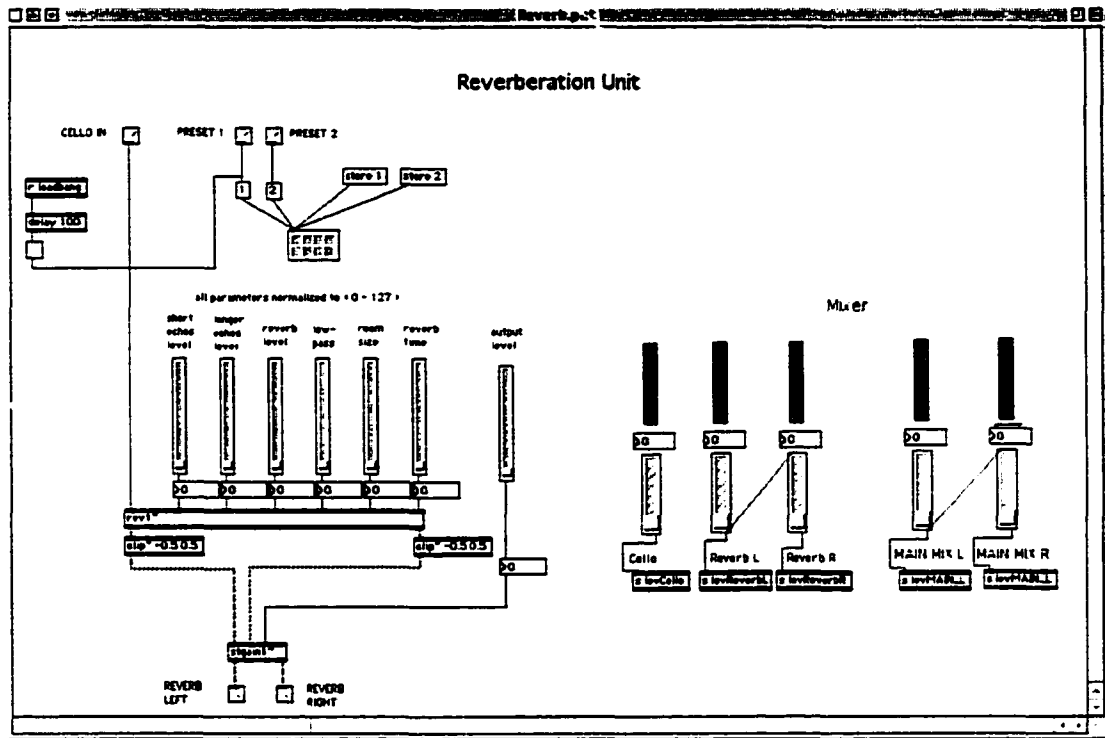


3.3.2 Audio Effect Processor Unit Design

While the need to select different patches during performance required two hardware audio processing units (Lexicon LXP-15 and Digitech TSR-24) in the original setup, the power of dynamic control structures allows real-time control for different program presets of digital audio signal processing, permitting only one software-based processing unit to give quite satisfactory results.

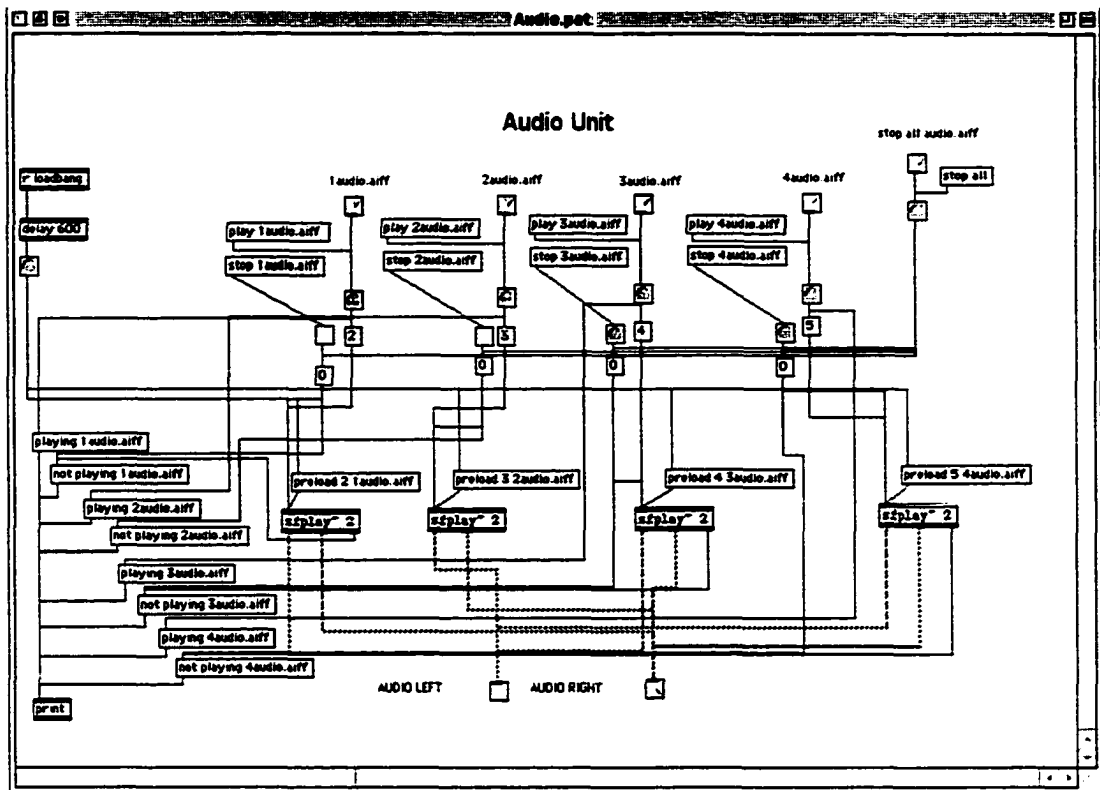
The audio effects processor unit design is based on the “rev1~” example developed by Zack Settel and provided with the “MSP Jimmies 1.0” MSP objects library [17]. This

patch provides different types of control over the input audio signal including short and long echo levels, room size, reverberation time, reverberation level and low pass filtering.



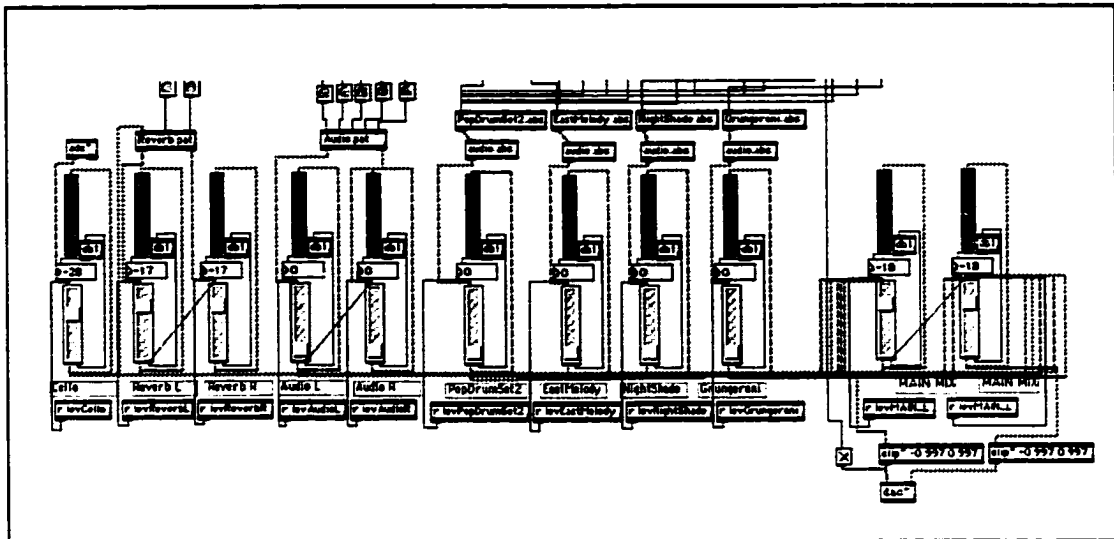
3.3.3 CD-ROM Drive

Although the built-in CD-ROM drive could be used for triggering CD-DA events, due to the unavailability of its audio level control within Max/MSP, the decision was made to play AIFF files from the computer hard drive instead, which provided fast access time and complete dynamic level control. The following example "Audio.pat" illustrates the use of the "sfplay~ 2" object to play AIFF files.



3.3.4 Mixing Board

The development of a software-based mixing board gives complete control over the final mix of four MIDI sampler instruments, input cello signal, audio effect processor and AIFF file playback, enabling the performer to have real-time control of all audio streams. The graphical support helps to visualize all the sound levels and the modified “db1” object [18] represents the gain scale, according to the digital audio standard. To avoid any possible clipping, during the final stage of DASP the “clip~” object was used.



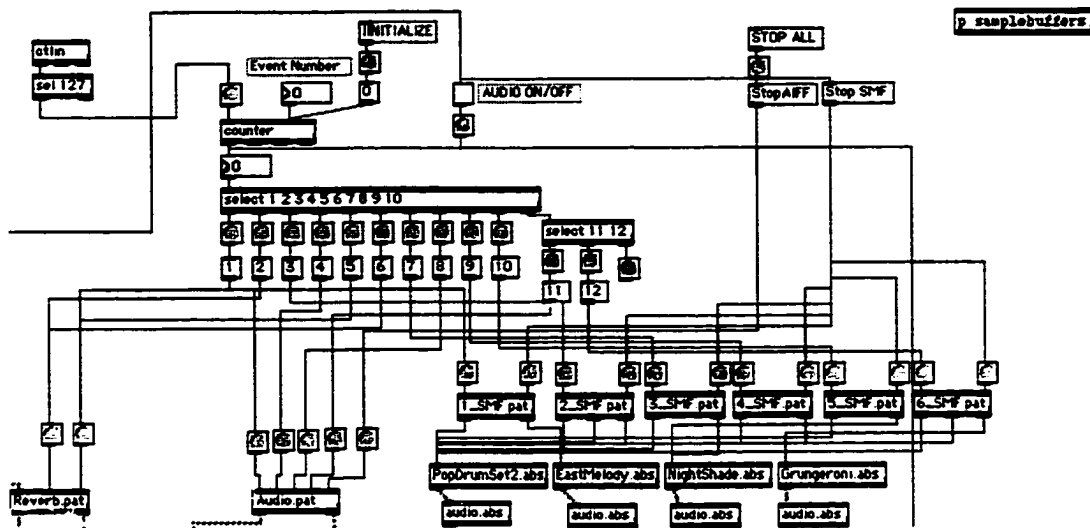
3.4 Dynamic System Control: Developing Control Structures

The issue of dynamic system control is one of the most important and critical questions in any real-time electro-acoustic system, including “Yonge Street Variations”. There are three types of real-time control structures in this software implementation:

- a) control of time-based events;
- b) control of MIDI based events;
- c) control of DASP events.

3.4.1 Control of Time-Based Events

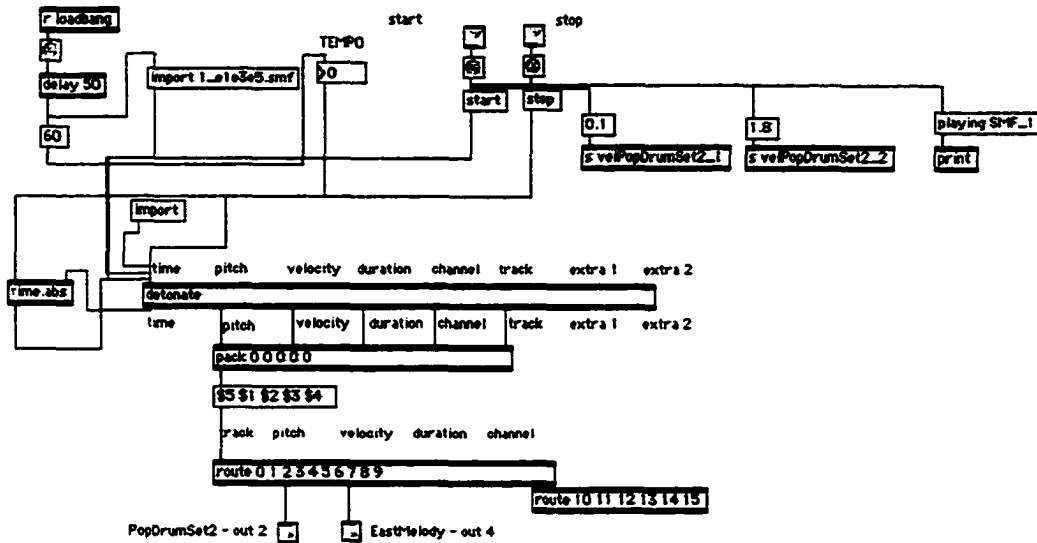
The first type is based on control structures for triggering fourteen different events with a foot-switch pedal, plugged in to a MIDI device connected to a computer via a standard MIDI interface. The design of this structure is based on the Max “counter” and “select” objects, and provides all the necessary tools for the desired level of control. For the convenience of the user, all the foot-switch triggering events are passed to the “ctlin” object without the need for the controller and channel numbers. From the “Yonge Street Master.pat” patch, any event can be triggered by either a MIDI input signal or a computer system event such as a mouse click, providing an additional level of control.



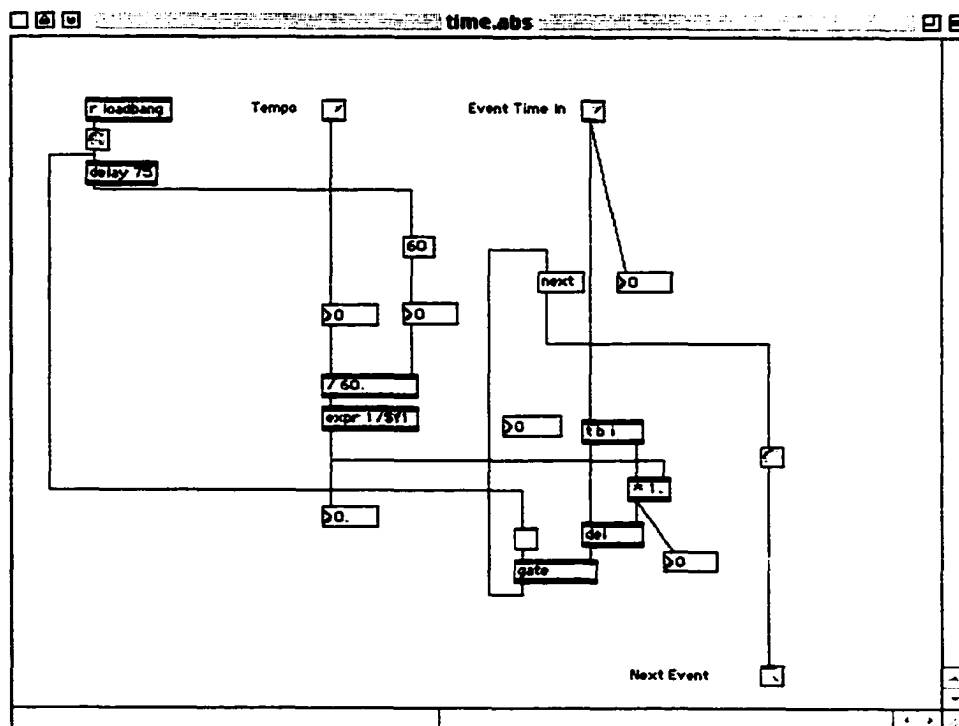
3.4.2 Control of MIDI based events

Two types of control for MIDI-based events can be found in the “Yonge Street Variations”: global MIDI based events such as playback of SMF files or tempo changes and local MIDI-based events such as additional velocity post-processing.

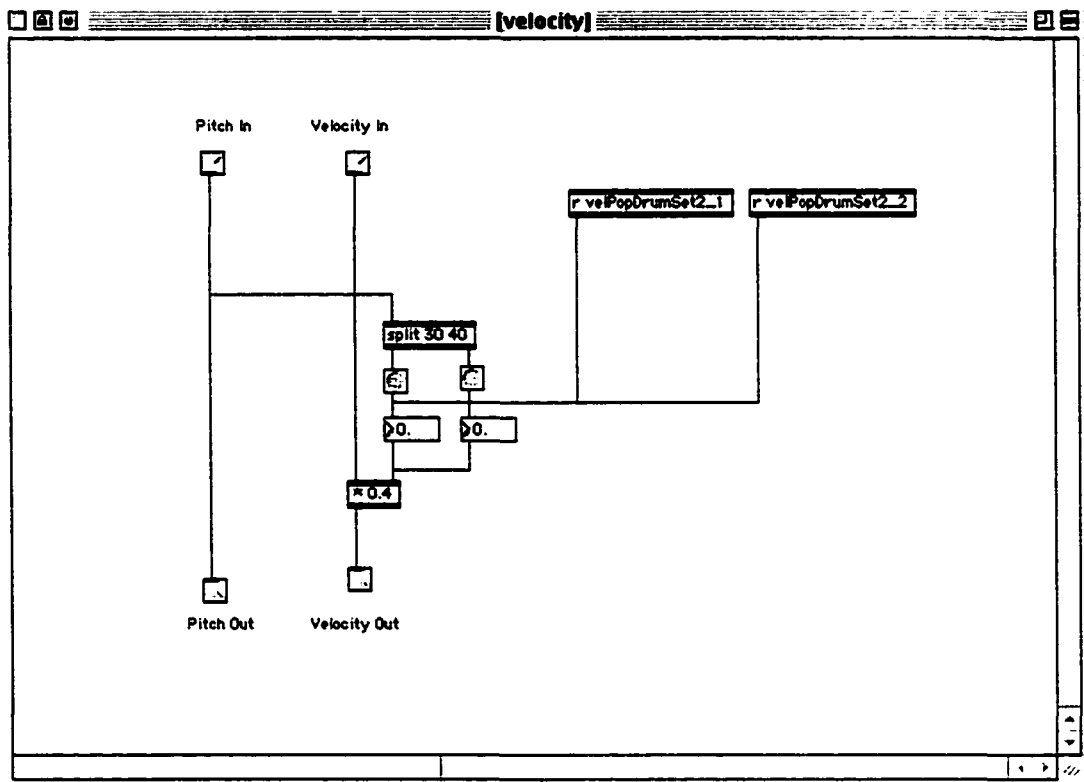
Due to the unavailability of the “play SMF” object used in the original software for the Apple Power Macintosh G3, a similar control structure was built based on the “detonate” object.



The global tempo control of the six SMF files is based on the following patch:



The local control of MIDI-based events includes velocity “normalization” of the “PopDrumSet2” sampler instrument due to its extremely wide dynamic range. All the control is performed via “send” and “receive” objects, which are triggered according to the SMF file number. All the processing is done by the small “velocity” sub-patch, which is located in the “PopDrumSet2.abs” abstraction.



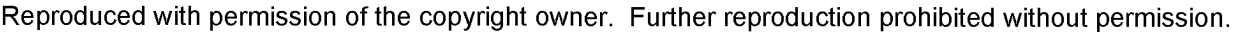
3.4.3 Control of DASP events

The real-time control of digital audio signal processing in the “Yonge Street Variations” electro-acoustic system includes the development of an audio effects processing unit, and mixing boards on the “master” and local level. The levels of all

audio inputs as well as the parameters of the effects processing unit are controlled in real-time as was discussed in the “Digital Audio Signal Processing: Designing a Software Implementation of the External Hardware” section.

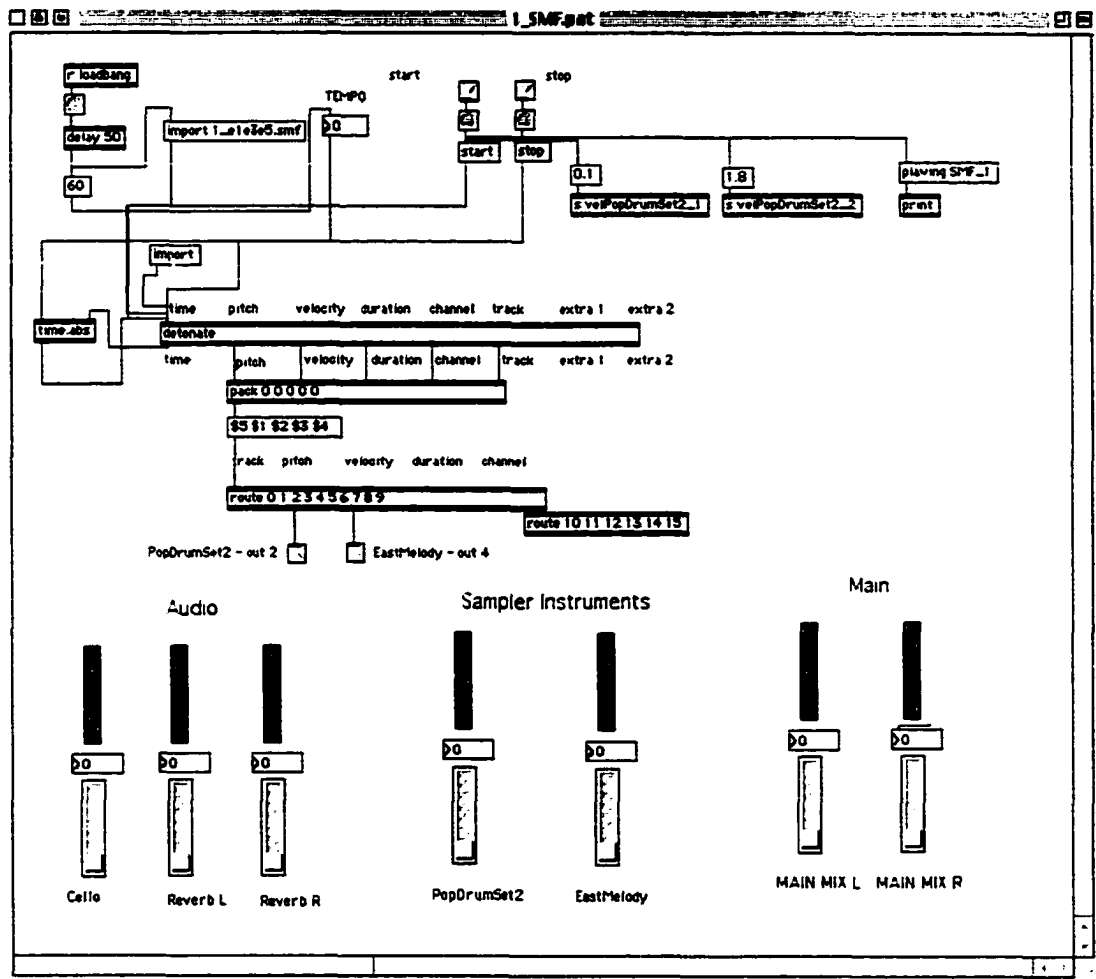
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

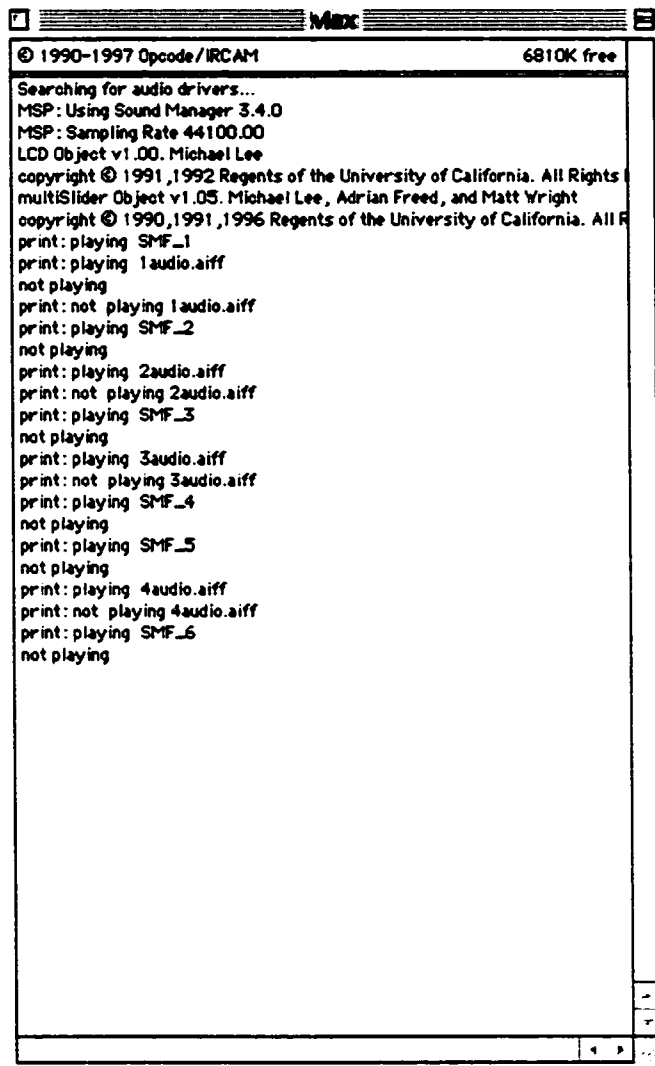


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

The following example of the GUI of the “sub-master” patch was designed to help the performer learn and rehearse the composition. It provides detailed control over the compositional structure, giving the possibility to perform and mix from the current event number.



For the convenience of the user, all the playback events, such as AIFF and MIDI files, are labeled and all the performed information may be observed during performance in the Max window.



3.6 CPU Performance and System Optimization

In order to minimize the performance requirements of “Yonge Street Variations” and the utilization of the computer central processing unit (CPU), the following system optimization was done:

1) All the sampler instrument voices were examined and according to the maximum number of notes sounding at one time the amount of polyphony for each voice was restricted as follows:

PopDrumSet2 - 11 notes polyphony,

NightShade - 6 notes polyphony;

EastMelody - 4 notes polyphony;

Grungeroni - 7 notes polyphony.

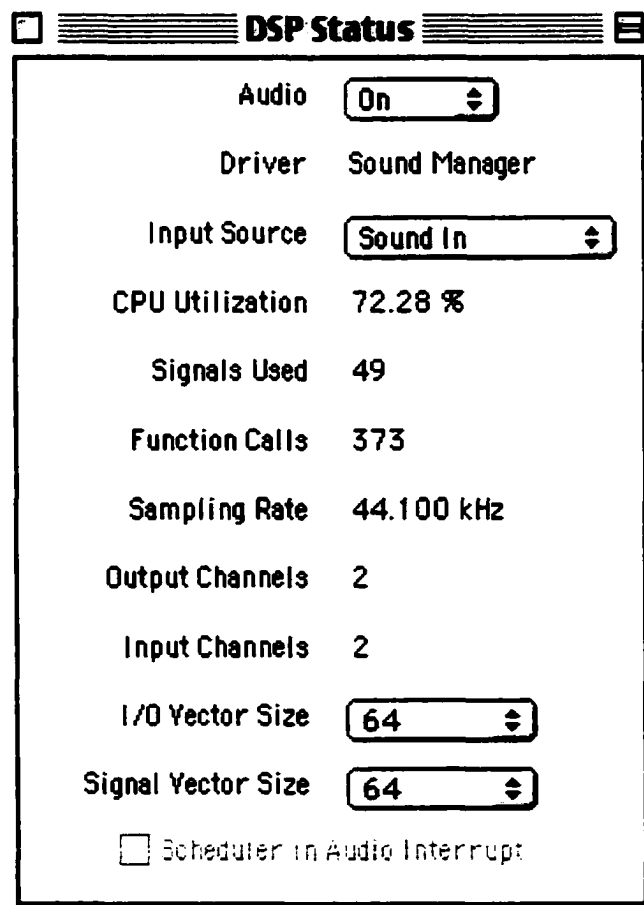
2) All the possible combinations of I/O Vector Size and Signal Vector Size were examined and the best quality of signal processing of the incoming audio stream was determined to be achieved with the following parameters:

Vector Size: 64;

Signal Vector Size: 64.

3.7 System Evaluation

The developed software real-time electro-acoustic music system for the performance of the “Yonge Street Variations” provides a prototype example of the software solution to the problem of portability and hardware dependence. The system design provides a stable performance environment utilizing from 65% to 75% (depending on the complexity of the patch) of the Apple Macintosh G3/300 MHz. This information can be observed in the DSP status window.



Although the designed system does not provide complete hardware independence (some of the hardware is still required, such as a MIDI device for the connection of foot-switch pedal, a MIDI interface, and a mixer or some microphone amplifier for the amplification of a certain type of microphones) most of the serious issues linked to the use of specific models of synthesizer or other audio signal processing devices are resolved.

4. Conclusion

Section two “Real-Time Electro-Acoustic Music Systems” provides the reader with some useful clarifications for the terminologies of the genre electro-acoustic music. These revised definitions and discussions of digital audio signal processing set the stage for the primary objective that is to embrace all these capabilities into a single unified electro-acoustic music delivery platform.

This examination of the Max/MSP programming language in the context of the development of a prototype example of a real-time electro-acoustic music system demonstrates the great potential of this programming environment for the field of live electro-acoustic music.

Providing a complete integration with the MIDI protocol, Max/MSP provides the possibility of the use of real-time DASP with the integration of various MIDI devices such synthesizers, samplers and many different types of controllers. All of these features combine to create a very valuable programming environment for the implementation in software of real-time electro-acoustic music systems.

With the introduction of Max/MSP, many of the historical issues related to live electro-acoustic music systems can be successfully resolved. These include such problems as prohibitive cost, portability and hardware dependence that are still relevant today. Although some issues such as cross-platform problems and operational system dependence still remain unresolved, it is likely that in the future further refinement and improvement will be done.

The introduction of the Max/MSP programming language as a cost-effective solution brings to the desktop the possibility of real-time digital audio signal processing and dynamic control structures in one integrated programming environment and makes it available for general use. The potential of Max/MSP is clear and in the near future it will likely continue to have a wide range of applications in the domain of digital audio.

References

- [1] Austin, Larry and Waschka, Rodney. Computer Music for Compact Disc: Composition, Production, Audience, In *Computer Music Journal*, 20 (2), pp.17-27, Cambridge, MA: MIT Press, 1996

- [2] Pennycook, Bruce. Live Electroacoustic Music: Old Problems, New Solutions, *Journal of New Music Research*, 26 (1), pp. 70-95, 1997

- [3] Lippe, Cort, Settel, Zack, Puckette, Miller, Lindemann, Eric. The IRCAM Musical Workstation: A Prototyping and Production Tool for Real-Time Computer Music, IRCAM, Paris, Manuscript

- [4] Dobson, Richard. A Dictionary of Electronic and Computer Music Technology: Instruments, Terms, Techniques, Oxford University Press Oxford, New York, 1992

- Cary, Tristram. *Dictionary of Musical Technology*, Greenwood Press, New York, 1992

- Tomlyn, Bo and Leonard, Steve. *Electronic Music Dictionary: A Glossary of the Specialized Terms Relating to the Music and Sound Technology of Today*, 1st edition, H. Leonard Books, Milwaukee, WI, U.S.A., 1988

- Anderton, Craig. *The Electronic Musician's Dictionary*, Amsco Publications, New York, 1988

- [5] Schrader, B. *Introduction to Electro-acoustic Music*, Prentice Hall, Englewood Cliffs, N.J., 1982, pp, 2-3

- [6] Dobson, Richard. **A Dictionary of Electronic and Computer Music Technology: Instruments, Terms, Techniques**, Oxford University Press, Oxford, New York, 1992, p.174

- [7] Rowe, Robert. **Interactive Music Systems: Machine Listening and Composing**. Cambridge, Massachusetts, London, England: MIT Press, 1993, p.1

- [8] Malham, D.G. **Spatial Hearing Mechanisms and Sound Reproduction**, University of York, England, 1998
http://www.york.ac.uk/inst/mustech/3d_audio/ambis2.htm

- [9] Ludeman, Lonnie C., **Fundamentals of Digital Signal Processing**, Harper & Row Publishers, New York, 1986

- Lynn, Paul A. **Introductory digital signal processing with computer applications**, 2nd ed., John Wiley & Sons Ltd., Chichester, New York, 1998

- Madisetti, Vijay K., Williams, Douglas B., edited by, **The Digital Signal Processing Handbook**, CRC Press, Boca Raton, IEEE Press, New York, 1998

- Mitra, Sanjit K., Kaiser, James F., edited by, **Handbook for Digital Signal Processing**, J. Wiley & Sons Inc., New York, 1993

- Paturzo, Bonaverta Antony. **Making Music with Microprocessors**, Blue Ridge Summit, PA, TAB Books Inc., 1984

- [10] Freedman, Alan. **Computer Desktop Encyclopedia**, American Management Association, New York, 1996, p. 262

- [11] Dodge, Charles and Jerse, Thomas A. *Computer Music: Synthesis, Composition and Performance*, Schirmer Books, 1997, p. 437
- [12] Roads, Curtis. *The Computer Music Machine*, Cambridge, Massachusetts, The MIT Press, 1989, p. 802
- [13] MIDI 1.0 detailed specification, International MIDI Association, Los Angeles, 1989
- [14] Dobrian, Christopher, *Max 3.5: Getting Started*, Opcode Systems, Inc. / IRCAM, 1990-1999, p. 4
- [15] What is MSP, Cycling '74 Home Page, Cycling '74, 1999
<http://www.cycling74.com/products/index.html>
- [16] Dobrian, Christopher, *MSP: The Documentation*, Cycling '74, 1998, p. 114
- [17] *Jimmies for MSP. Handbook*, First English edition, IRCAM, Centre Georges Pompidou, Paris, 1998, p. 32
- [18] *Jimmies for MSP. Handbook*, First English edition, IRCAM, Centre Georges Pompidou, Paris, 1998, p. 32

Bibliography

Anderton, Craig. *The Electronic Musician's Dictionary*, Amsco Publications, New York, 1988

Austin, Larry and Waschka, Rodney. *Computer Music for Compact Disc: Composition, Production, Audience*, In *Computer Music Journal*, 20 (2), pp.17-27, Cambridge, MA: MIT Press, 1996

Baggi, Denis, edited by. *Computer-Generated Music*, IEEE Computer Society Press, Los Alamitos, California, 1992

Chadabe, Joel. *Electric Sound: The Past and Promise of Electronic Music*, Upper Saddle River, New Jersey, Prentice Hall, 1997

Dannenberg, Roger B. *A Perspective on Computer Music*. In *Computer Music Journal*, 20, pp.52-56, Cambridge, MA: MIT Press, 1996

Deutsch, H. *Electroacoustic Music – The First Century*, Miami, Belwin Mills, 1993

Dobson, Richard. *A Dictionary of Electronic and Computer Music Technology: Instruments, Terms, Techniques*, Oxford University Press, Oxford, New York, 1992

Dobrian, Christopher, *Max 3.5: Getting Started*, Opcode Systems, Inc. / IRCAM, 1990-1999

Dobrian, Christopher, *Max 3.5: Reference*, Opcode Systems, Inc. / IRCAM, 1990-1999

Dobrian, Christopher, *MSP: The Documentation*, Cycling '74, 1998

Dodge, Charles and Jerse, Thomas A. Computer Music: Synthesis, Composition and Performance, Schirmer Books, 1997

Fluckinger, Francois. Understanding Networked Multimedia Application and Technology, Prentice Hall, 1995

Freedman, Alan. Computer Desktop Encyclopedia, American Management Association, New York, 1996

Haus, Goffredo, edited by. Music Processing, A-R Editions, Inc., Madison, Wisconsin, 1993

Holmes, Thomas B. Electronic and Experimental Music, New York, Charles Scribner's Sons, 1985

Horn, Delton T. The Beginner's Book of Electronic Music, Blue Ridge Summit, PA, TAB Books Inc., 1982

Jacobs, Gabriel and Georgiades, Panios, Music and New technology – the MIDI connection, Sigma Press, Wilmslow, England, 1991

Jimmies for MSP. Handbook, First English edition, IRCAM, Centre Georges Pompidou, Paris, 1998

Lippe, Cort, Settel, Zack, Puckette, Miller, Lindemann, Eric. The IRCAM MusicWorkstation: A Prototyping and Production Tool for Real-Time Computer Music, IRCAM, Paris, Manuscript

Kennedy, Michael edited by. The Oxford Dictionary of Music, 2nd edition, Oxford University Press, Oxford, New York, 1994

Lambert, Steve and Sallis, Jane, edited by. CD-I and Interactive Videodisk Technology, Howard W. Sams & Co., 1987

Ludeman, Lonnie C., Fundamentals of Digital Signal Processing, Harper & Row Publishers, New York, 1986

Lynn, Paul A. Introductory Digital Signal Processing with Computer Applications, 2nd ed., Chichester, New York: John Wiley & Sons Ltd., 1998

Malham, D.G. Spatial Hearing Mechanisms and Sound Reproduction, University of York, England, 1998
http://www.york.ac.uk/inst/mustech/3d_audio/ambis2.htm

Madisetti, Vijay K., Williams, Douglas B., edited by, The Digital Signal Processing Handbook, CRC Press, Boca Raton, IEEE Press, New York, 1998

Manning, Peter. Electronic & Computer Music, 2nd ed., Clarendon Press, Oxford, 1993

MIDI 1.0 detailed specification, International MIDI Association, Los Angeles, 1989

Mitra, Sanjit K., Kaiser, James F., edited by, Handbook for Digital Signal processing, J. Wiley & Sons Inc., New York, 1993

Moore, Richard F. Elements of Computer Music, Englewood, Cliffs. NJ, PTR Prentice Hall, 1990

Opcode's Max, Opcode Systems Home Page, Opcode Systems Inc., 1999

<http://www.opcode.com/products/max/>

Paturzo, Bonaverta Antony. *Making Music with Microprocessors*, Blue Ridge Summit, PA, TAB Books Inc., 1984

Pennycook, Bruce. *Live Electroacoustic Music: Old Problems, New Solutions*, *Journal of New Music Research*, 26 (1), pp. 70-95, 1997

Pennycook, Bruce. *Language and Resources: A New Paradox*, In *The Language of Electroacoustic Music*, Macmillan Press, Music Division, 1986

Pierce, John R. *Computer Music, Coming and Going*, In *Computer Music Journal*, 20 (1), pp.49-51, Cambridge, MA: MIT Press, 1996

Pohlamn Ken C. *Principles of Digital Audio*, 2nd ed. SAMS, 1990

Pohlamn Ken C. *The Compact Disc Handbook*, 2nd ed. Madison, Wisconsin: A-R Editions, Inc., 1992

Pope, Stephen Travis, edited by. *The Well-Tempered Object. Musical Applications of Object-Oriented Technology*, The MIT Press, Cambridge, Massachusetts, London, England, 1991

Proakis, John G., *Introduction to Digital Signal Processing*, Macmillan Publishing Company, New York, Collier Macmillan Publishers, Macmillan, 1988

Randel, Don Michael edited by. *The New Harvard Dictionary of Music*, The Belknap Press of Harvard University Press, Cambridge, Massachusetts, 1996

Roads, Curtis, edited by. *Composer and the Computer*, Los Altos, California, William Kaufmann, Inc, 1985

Roads, Curtis. *The Computer Music Machine*, Cambridge, Massachusetts, The MIT Press, 1989

Roads, Curtis, edited by. *The Computer Music Tutorial*, Cambridge, Massachusetts, The MIT Press, 1996

Rothstein, Joseph. *MIDI A Comprehensive Introduction*, 2nd ed. Madison, Wisconsin: A-R Editions, Inc., 1995

Rowe, Robert. *Interactive Music Systems: Machine Listening and Composing*. Cambridge, Massachusetts, London, England: MIT Press, 1993

Russ, Martin. *Sound Synthesis and Sampling*, Focal Press, 1996

Schrader, B. *Introduction to Electro-acoustic Music*, Englewood Cliffs, N.J., Prentice Hall, 1982

Schwartz, Elliott. *Electronic Music: A Listener's Guide*, revised edition, Da Capo Press, New York, 1989

Tomlyn, Bo and Leonard, Steve. *Electronic Music Dictionary: A Glossary of the Specialized Terms Relating to the Music and Sound Technology of Today*, 1st edition, H. Leonard Books, Milwaukee, WI, U.S.A., 1988

Warner, Josh. *The Enhanced CD Fact Book*, Version 2.0, Apple Computer, Inc., 1997

Watkinson, John. *An Introduction to Digital Audio*, Focal Press, 1994

Watkinson, John. The Art of Digital Audio, Focal Press, London & Boston, 1988

What is MSP, Cycling '74 Home Page, Cycling '74, 1999

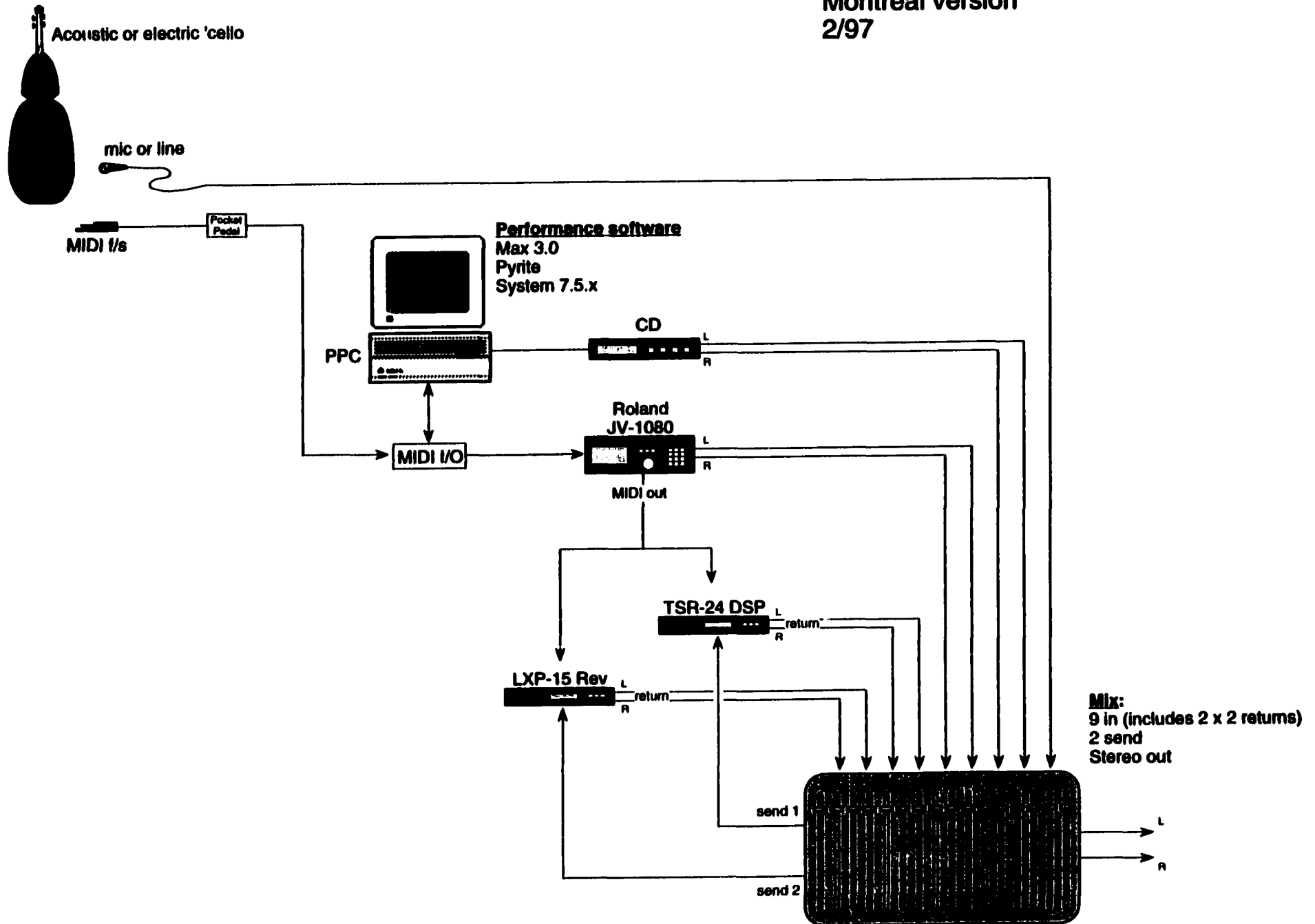
<http://www.cycling74.com/products/index.html>

The Yonge Street Variations

System Setup

The Yonge Street Variations

Montreal version
2/97



The Yonge Street Variations

Score

Yonge Street Variations

Bruce Pennycook

First system of the score, measures 1 through 2.

Percussion: Measures 1 and 2 are marked with a box labeled **e1** and a box labeled **e2** respectively. Measure 1 has a box labeled **6** above it. Measure 2 has a box labeled **1** above it. Measure 3 has a box labeled **2** above it.

System: Measure 1 has a box labeled **DSP** above it.

Violoncello: Measure 1 has a box labeled **pizz.** above it. Measure 2 has a box labeled **open** above it. Measure 3 has a box labeled **arco sul pont.** above it. Measure 4 has a box labeled **very brittle** above it. Measure 5 has a box labeled **fff** below it. Measure 6 has a box labeled **sfp** below it. Measure 7 has a box labeled **fff** below it.

Other markings include **L.v.** (Larghetto) and **fff** (fortissimo).

Second system of the score, measures 3 through 7.

Perc. (Percussion): Measures 3 through 7 are marked with boxes labeled **3**, **4**, **5**, **6**, and **7** respectively.

Sys. (System): Measures 3 through 7 are marked with boxes labeled **3**, **4**, **5**, **6**, and **7** respectively.

Vcello (Violoncello): Measures 3 through 7 are marked with boxes labeled **3**, **4**, **5**, **6**, and **7** respectively. Measure 3 has a box labeled **mf** below it. Measure 4 has a box labeled **p** below it. Measure 5 has a box labeled **pp** below it. Measure 6 has a box labeled **ff** below it. Measure 7 has a box labeled **(nat.)** above it.

Other markings include **mf** (mezzo-forte), **p** (piano), **pp** (pianissimo), and **ff** (fortissimo).

Third system of the score, measures 8 through 11.

Perc. (Percussion): Measures 8 through 11 are marked with boxes labeled **8**, **9**, **10**, and **11** respectively.

Sys. (System): Measures 8 through 11 are marked with boxes labeled **8**, **9**, **10**, and **11** respectively.

Vcello (Violoncello): Measures 8 through 11 are marked with boxes labeled **8**, **9**, **10**, and **11** respectively. Measure 8 has a box labeled **sfp** below it. Measure 9 has a box labeled **pp** below it. Measure 10 has a box labeled **f** below it. Measure 11 has a box labeled **sfp** below it. Measure 12 has a box labeled **ff** below it.

Other markings include **sfp** (sforzando), **pp** (pianissimo), **f** (forte), and **ff** (fortissimo).

12 13

Perc.

Sys.

Vcello

3 3 6 *fp* < >

14 15

Perc.

Sys.

Vcello

6 6 fade

p *f* *fp* 6

16 17

Perc.

Sys.

Vcello

6 L.H. L.H. pizz. *fp* 3 6

18 19

Perc.

Sys.

Vcello

6 *tr* (b) *tr* (b) 6 3

e4

20 21 22 23

Perc.

Sys. **DSP**

Vcello

long rev / ch

f *mp* *p*

e5

24 25 26

Perc.

Sys.

Vcello

rep / vary Mm. 23,24

fp

e6

27 28 29 30

Perc.

Sys. *molto sus*

Vcello

art.

e7

31 32 33 = 50

Perc.

Sys. **DSP**

Vcello

ff

9 -3 9 9

34 35

Perc.

Sys.

Vcello

3 3 2 3 9 3 3

ff

36

Perc.

Sys.

Vcello

3 9 3

37 38

Perc.

Sys.

Vcello

3 3 3 3 9

fp *f*

39 40

Perc.

Sys.

Vcello

3 3 3 3 3 3 3 3

f *mp*

41 42

Perc.

Sys.

Vcello

cresc. -----

fff *dim.* -----

43 44 45 46

Perc.

Sys.

Vcello

p

3

3

9

47 48 49

Perc.

Sys.

Vcello

3

5

3

5

sfz

sfz

50 51 52

Perc.

Sys.

Vcello

mf

mp

pizz *p*

sfzp

spic. 9

spic. 3

3 3

sus

rit. ----- *a tempo*

rit. -----



53 54 55 56 57

Perc.

Sys.

Vcello

p

sul pont.

art.

art. 8^{va}



58 59

Perc.

Sys.

Vcello

fff *sfz* *dim.* *pp* *a niente*

9 9 9

sim, falls apart



60 61

Perc.

Sys.

Vcello

10 - 20"

DSP

improvise

very high, art./nat harmonics, long notes but varied double stops, overlaps

p

fade with system

♩ = 60 basic pulse

62 63

Perc.

Sys.

Vcello

mf

sus, sempre

6 6

3 3 3

8^{va}

64 65 66

Perc.

Sys.

Vcello

gliss.

3

3

3

3

f dim.

nat.

67 68 69

Perc.

Sys.

Vcello

nat.

pp

f

3

3

3

70 71 72 73

Perc.

Sys.

Vcello

3

gliss.

3

p

pp

74 75 76 77 78

Perc.

Sys.

Vcello

3

wide vib, 1 note only

79 80 81 82 83

Perc.

Sys.

Vcello

no vib

sul pont. no vib

sfzp *f* *p* *pp* *ppp*

84 soft noises

Perc.

Sys.

Vcello

DSP

Improvise:
(drum brushes)
rapid, nervous,
irregular on pitches,
open strings, body
above bridge etc.

p

match intensity to tape

85

Perc.

Sys.

Vcello

cresc.

fff

DSP

greatly increase everything

sim

fff

86

Perc.

Sys.

Vcello

freely, quasi guitar

pizz

sim

3

5

arco

very slow gliss.

strum

ff *p* *mp* *L.v.* *f* *f* *p* *ff*

87

Perc.

Sys.

stump, furiously
accel. -----

sim

ff *ff* *pp* *p*

88

Perc.

Sys.

pizz. 3 5 arco strum sim

mf *f* *f* *pp* *p* *ff* *ff*

89

Perc.

Sys.

arco accel. -----

sfzp *sfzp* *sfzp* *sfzp* *sfzp* *sfzp* *sfzp*

Perc.

Sys.

accel. ----- molto ----- highest sul pont. micro

sfzp *ff* *dim.* *ppp* *ppp* *pp* *pp*

90

Perc.

Sys.

Vcello

ord. arco

pizz.

rit.

arco accel.

f *p* *ff* *mf* *ff* *fff*

91 92 93

Perc.

Sys.

Vcello

evenly strum

ritard

repeat

p

a niente

94 = 70 95 96

Perc.

Sys.

Vcello

strict tempo

pizz.

arco

pizz.

trill

arco

pizz.

mf *p*

97 98 99

Perc.

Sys.

Vcello

arco

pizz.

arco

p *f* *p* *mf* *f* *ff*

[isorhythmic pulsing]

100 101 102

Perc.

Sys.

Vcello

arco

tr

tr

pizz.

arco sul pont.

103 104 105

Perc.

Sys.

Vcello

f

p

loco spic.

106 107

Perc.

Sys.

Vcello

DSP

spic.

pp

Improvise

repeat freely after tempo but not pitch; always lightly

(trance-like)

sim

Perc.

Sys.

Vcello

long fade with system (reduce swells) . . .

sim

sim

sim

sim

ppp

freely
108

Perc.

Sys.

only pitch set

body slap

wait for rev.

ad lib from pitch set (suggested)

(2 hands) free pitches very rapidly (erratic)

Vcello

pizz.

p

fff

fff

109

Perc.

Sys.

wait for rev.

2 hand pizz. (sim)

2 hand (sim)

Vcello

fff

fff

pp

fff

pp

Perc.

Sys.

wait for rev

(pitch set) rapidly, not even

pizz.

sim ad lib

Vcello

f

fff

fff

p

Perc.

Sys.

sim

become less even

pizz.

arco detune

Vcello

fff

p

fff

fff

sfzpp

Perc.

Sys.

Vcello

mf *sim* *fff* *sfz* *fff* *p*

furious, brief 2 hand *pizz. (rasq.)*

Perc.

Sys.

Vcello

fff *dim.*

2 hand (very erratic)

Perc.

Sys.

Vcello

p *mf* *p*

behind *behind arco*

Perc.

Sys.

Vcello

fff *p* *p* *p*

with rev. *sul pont. nat.* *(blend with rev / delay)* *[nearly even ♩ = 40]*

Perc.

Sys.

up A natural harms.

Vcello

niente

110 111 112

Perc.

Sys.

furioso

Vcello

ff *ff* *fp*

113 114 115

Perc.

Sys.

Vcello

tr (#)

step <>

f

116 117 118

Perc.

Sys.

Vcello

sfz p

119 120

Perc.

Sys.

Vcello

fp *fp*

121 122 123

Perc.

Sys.

Vcello

mf *mp* *p* *p*

124 125 126 127

Perc.

Sys.

Vcello

p *pp* *pp* *p* *sfz* *p*

128 129 very forcefully

Perc.

Sys.

Vcello

3 5

130 131 132

Perc.

Sys.

Vcello

3 *ff* \lessgtr

133 134

Perc.

Sys.

Vcello

6 6 3 6

135 136

Perc.

Sys.

Vcello

3 \lessgtr *p* *f* 6 6 6

137 138

Perc.

Sys.

Vcello

3 6 3 *f* *ff* *fp* \lessgtr *ff*

139 140

Perc.

Sys.

Vcello

141

Perc.

Sys.

Vcello

142 143

Perc.

Sys.

Vcello

144 145 146 147

Perc.

Sys.

Vcello

fp

gliss.

Technical Requirements

Hardware

Apple Power Macintosh G3/300MHz/192 RAM

Foot-Switch Pedal

MIDI Interface

MIDI device

MIDI cable

Microphone (preferably cardioid , condenser)

Microphone Amplifier or Mixer (if necessary)

Software

MacOS 8.5

Opcode Max 3.5

Cycling'74 MSP

Opcode OMS 3.6.2

CD-ROM Content

I. “Yonge Street Variations” EAMS Software

II. Resources

1. YS Software
2. YongeStreetNotes
3. YS Documentation
 - a) Score
 - b) Setup.illustrations
 - c) Fonts

Software required to read this CD-ROM:

MacOS 8.5

Opcode Max 3.5

Cycling'74 MSP

Opcode OMS 3.6.2

Microsoft Word 98

Finale 3.5

The Yonge Street Variations
CD-ROM