

# 1. Introduction to the ECMC Computer Network

(This section last updated August 2007)

The purpose of this introductory section is to assure that all users of the ECMC advanced computer music studio are familiar with certain basic concepts and terms relating to computer systems and to *digital audio workstations* (computer systems used to make music). You may already know much of this material, and thus may find portions of this section to be a wearisome or tedious read. Be patient, and if necessary read this material while brushing your teeth or folding laundry. Things quickly will become more challenging.

## 1.1. Computer Systems : Hardware and Software

A computer system is a combination of

- 1) physical equipment, called *hardware*
- and
- 2) sets of coded instructions that are executed by the system

These instructions are collectively called *software*, because they can easily be changed, updated or replaced — sometimes as often as several times a year — and because they can be copied for use on another computer system with compatible hardware and software.

A computer has a fairly limited range of basic capabilities, such as adding, subtracting, multiplying, dividing, comparing numbers, and keeping track of where information is stored. However, it can perform these elementary tasks with blinding speed. Each computer operation takes a small but finite amount of time, measured in microseconds (millionths of a second) or nanoseconds (billionths of a second). Computer hardware provides the raw capabilities of a system, but can do nothing by itself. Software combines elementary machine cycle operations (such as reading, writing and comparing data) into more complex instruction sets that accomplish particular tasks.

### Operating Systems

The most important piece of software of a computer system — the "master program" — is the **operating system** (*OS*). Operating systems regulate the processing, flow and storage of data, communicate with peripheral devices (such as disks, tape drives and printers), and perform lots of other functions, many of which are unseen by the user. The core of an operating system — the code that controls its most basic functions and capabilities — is fairly small and is called the *kernel*. Built on top of the kernel is a much larger layer of additional software that enable us to issue instructions, regulate the flow of data between various devices connected to the computer, and run applications.

#### *Operating systems in common use today*

- (1) Microsoft's *Windows NT*-based operating systems, including *Windows Vista* (the current incarnation of *Windows*), *Windows XP*, *Windows 2000* and *Windows 2003 server*.
- (2) Unix-based systems, including
  - ☞ *Macintosh* operating systems beginning with *Mac OS X*, in which Macintosh graphical tools and conventions have been grafted onto a Unix core
  - ☞ other proprietary, commercial variants of Unix from vendors such as Sun Microsystems (*Solaris*), IBM (*AIX*) and Hewlett-Packard (*HP-UX*)
  - ☞ open source Unix clones developed primarily by volunteers and distributed at low or no cost, including and *GNU Linux*, *FreeBSD* and *OpenSolaris* (an open source project sponsored by Sun)

Older operating systems, such as Microsoft's *Windows 98* and *ME* and all versions of the Macintosh *Finder* up through version 9.x, were designed to support a single user performing one task at a time, or, at best, one user switching back and forth between a few tasks. Such legacy systems, which are rapidly disappearing from desktop systems because of their limitations and lack of stability, are termed *single user, cooperative multitasking*. A program can "take over" the computer and can only be interrupted at times when the program "cooperates," or "agrees" to relinquish control of the system.

By contrast, Unix, an operating system originally developed at AT & T Bell Labs in the 1970s, Unix offshoots such as GNU/Linux,<sup>1</sup> Free BSD and Mac OS X, as well as versions of *Windows* since the introduction of *NT* in 1993, which incorporate many features similar to those found in Unix, are termed *multiuser*, *pre-emptive multitasking* systems. These systems can service several users simultaneously. Each user, in turn, may be working simultaneously on two or more concurrent processes. Unless a program is badly written, the operating system itself can interrupt and terminate programs when it senses that they have gone awry. This capability comes at some cost, however, as individual programs may run somewhat slower in a multitasking environment in which they are interrupted more frequently. However, given the speed of today's computer processors, and the growing use of computers that employ two or more processors, this is rarely an issue.

Operating systems that can support several simultaneous tasks, and two or more simultaneous users, generally employ *time sharing* procedures. Each process shares a portion of the computer's time. Jobs running at a higher system priority receive larger time slices than lower priority jobs, and system administrators can adjust the priorities of various types of processes. Currently, a majority of computers today actually can only work on one process at a time. Thus, most time-sharing computer systems actually are not working on several jobs simultaneously, but, rather, are switching back and forth between these processes. In many cases the computer is fast enough so that each user feels as though (s)he has the undivided attention of the processor. However, computationally or I/O intensive tasks (including many video and audio applications) can place sufficiently heavy demands upon the processor, bus and disks that the system may not respond as quickly as we would like or need to user input. This time delay, or *latency*, is a critical issue in audio and video applications.

To mitigate processing bottlenecks most higher end computers today incorporate two, four or more processors running *SMP* ("symmetrical multiple processor") software. *SMP* systems actually can work on two or more tasks simultaneously, by divvying up a complex task or multiple tasks into subroutines that run concurrently on separate processors.

*Pre-emptive multitasking* means that the operating system (and individual users) can issue halt instructions to programs while they are running. Thus, if a program goes haywire and locks up, a user can issue an interrupt command and kill the rogue program, and need not perform a system shut down and reboot in order to regain control of the system.

Most Microsoft *Windows* systems, which currently dominate more than 90 % of the desktop (small computer) market, run on *Pentium* model processors manufactured by the Intel Corporation, or else on processors such as the *Athlon* models manufactured by the AMD (Advanced Micro Devices) Corporation. Such computers often are referred to simply as *PCs* ("Personal Computers"). More accurately, however, a *PC* is simply a "desktop" ("one-user-at-a-time") workstation, as opposed to

- a larger, more powerful *server*, or network of servers, which service requests for data or applications from client computers; a large web site such as *Google* employs thousands of servers that access a common database
- a portable *laptop* or *notebook* system;
- a palm or hand-held PDA ("personal digital assistant") device; or
- an *embedded* ("hidden from the user") operating system controlling some electronic device, such as a cellular phone or digital camera.

When several users are logged onto a multiuser computer system and submitting jobs, or when one user makes intensive demands on a system (such as by loading many very large soundfiles into memory simultaneously), it is possible that there will not be enough RAM (computer memory) to store all of the

---

<sup>1</sup> Technically, the term "*Linux*" refers only to an operating system *kernel* developed by Linux Torvalds and a team of Linux Project programmers. The *GNU* (pronounced "*ga-NOO*") *Project* is a non-profit organization that supports the development of a broad range of open source applications, without which the Linux kernel would be useless. Many users of GNU software on Linux kernels prefer to call their operating system *GNU Linux*. In common usage, however, the term *Linux* often is used loosely to refer to all GNU operating system components running on a Linux kernel. During the past few years usage of GNU Linux software has grown very rapidly, and GNU software is rapidly replacing more expensive, closed source proprietary versions of Unix.

For more information on the history of GNU software, the Linux kernel, the *GPL* (General Public License) under which GNU software is distributed, and contrasts between open and closed source software paradigms, see <http://www.gnu.org/gnu/thegnuproject.html> or <http://web.dodds.net/~hozer/dream.html>.

programs and their data simultaneously. When this happens, the OS "swaps out" (copies) the current version of user A's program, which is not being executed at that instant, to a swap area on a system hard disk. As soon as the operating system decides that it is time to return to user A's program, the processor swaps out someone else (or else swaps out another of user A's processes) and copies user A's program back into RAM. All of this generally is accomplished automatically, by memory management programs within the operating system.

For many users, one of the most important elements that determines their preference for a particular operating system is its **user interface** (sometimes also called a "*desktop environment*"). A graphical, mouse-based *window manager* helps users to find, launch and control program applications by means of icons, menus and other graphical symbols, to open multiple windows, each used for a different process or purpose, to move back and forth between these windows (and the processes they represent), and, sometimes, to "draw" (rather than type in) input data. Such graphical tools can make applications more intuitive, easier to learn and to negotiate, but they also increase system overhead, slowing down many tasks. Additionally, it sometimes is much quicker to type a few characters to find a file, to launch a task or to perform an operation rather than to negotiate your way through a series of chatty dialog boxes or exploding menus.

The software library of a computer system contains many *programs*. A **program** consists of a series of *commands* (instructions), and, often, of *data* to be processed. A simple program might tell a computer processor to add a column of numbers and display the result on a monitor. A complex program, such as a sophisticated sound synthesis algorithm, might include thousands or even millions of individual instructions (many of which may be repetitive), and might launch many subroutines or other programs necessary to complete the main program.

A program is often worked out initially as an **algorithm** — a generalized sequence of procedures that might be expressed in terms of a graphic diagram or in plain English. Algorithms are often likened to recipes or formulas. "Classical" frequency modulation, in which the frequency of an audio oscillator is varied periodically by another audio signal to produce either vibrato or a more complex timbre, is an example of an algorithm. A *program* is a specific implementation of an algorithm, written by means of the conventions and tools provided by a particular programming language. A program can be written (1) in a particular compiler language, such as *C*, *Lisp*, *Pascal* or the music compiler *Csound*, or (2) by combining a series of general purpose utility operations provided by the operating system, or else (3) in a graphical programming environment, such as the musical applications *MAX* and *PD*.

## 1.2. Data Representation; ASCII and binary files

Computers deal only with numbers. When we type in a command on a computer terminal keyboard, the terminal converts what we type into *ASCII* code, a digital representation of alphabetical, numerical and special characters.<sup>2</sup>

Using a *text editing* (or *word processing*) program, we can type information or data into an *ASCII file* (document) that is stored on a disk connected to the computer.<sup>3</sup> An *ASCII file* might contain a love letter or a shopping list. However, it might instead contain a program — a series of commands that we wish to execute in sequence. In Unix-based operating systems, many shorter *ASCII* program files are called *shell scripts*, because they are interpreted and executed by a program called a *shell*, which includes many built-in commands. The script file includes keyword commands that the shell recognizes and can execute. Cross-platform programming and scripting languages such as *Java*, *Perl* and *Python* sometimes work in a manner similar to Unix shell scripts, but allow the script to be run on computers employing different operating systems.

A shell, *Perl* or *Python* script runs comparatively slowly, and such scripts generally are used only for fairly simple tasks. Complex programs, especially those involving repetitive operations, run faster if they are coded into the machine language that is unique to the architecture of a particular type of

---

<sup>2</sup> Examples of "special" characters include \* (asterisk), # (pound sign), / (slash) and \ (backslash), the *space* character, the *control*, *alt*, *delete* and *escape* characters, and function keys.

<sup>3</sup> However, by default many word processing programs, such as Microsoft's *Word* (and its baby sibling *Wordpad*), do not save text input in *ASCII* format, but rather in a proprietary format that can only be read by the application and by compatible applications.

computer.

A *compiler* is a large program that converts ASCII code (instructions) into executable machine code. The programmer edits an ASCII *source code* file with a text editor, and then runs this source file through the compiler to produce an executable machine code file. The executable *binary* data in this machine code file cannot be viewed directly. Source code files can be copied from one computer to another, but must be compiled into an executable binary file on the new system before they can be used. Many types of raw data processed by computers, including the representation of sounds and video images, also are stored in binary (non-viewable) rather than ASCII format. The most commonly used compiler language is *C*, which comes in various flavors (e.g. *C++*, the open source *GCC* and Microsoft's *C#*).

The term *application* refers to a bundled package of one or more related executable (binary) programs and a *graphical user interface (GUI)* through which the user can control the program(s) with mouse movements and keyboard input. An application can be either commercial (developed and sold by a software company for profit), freeware (distributed at no cost) or shareware (distributed at nominal cost, often between \$20 and \$100).

Actually, the term *free software* has two distinct meanings:

- "free beer" : the software is available at no cost; and
- "free speech," or *open source* software : the compiled binary software, whether provided at no cost or sold for profit, is provided along with complete source code. Users can look at this source code to determine exactly how a program is working, and can alter and re-compile the program in order to fix bugs or to custom tailor the program to their own needs or preferences.

Microsoft's *Windows*, *Office* and other software, the Macintosh OS, commercial distributions of Unix and commercial musical applications such as *Cubase*, *Logic* and *Max/MSP* are examples of proprietary software. Users purchase the software for use on a single machine, or purchase a site license to run the software on a given number of machines, or for a specific number of individual users. Various protection schemes, such as hardware dongles (yuck!) or challenge-and-response codes are employed to protect the software from being installed and used simultaneously on several systems. Some vendors employ even more restrictive licensing schemes in which users "lease" rather than "buy" software, and must commit to future upgrade fees in order to keep the software fully functional and up-to-date.

Open source software has none of these restrictions. A majority of open source software, including GNU Linux and most of the music software available on the ECMC Linux systems, as well as some programs that also run on Windows and Macintosh systems (e.g. *Firefox*, *Csound* and *Audacity*) is licensed for use under some variant of the *General Public License (GPL)*,<sup>4</sup> which gives users broad rights to use, alter and redistribute the software.

However, some "no cost" ("free beer") software is proprietary rather than open source. Binaries to the Csound score pre-processor *score11*, for example, can be obtained from Aleck Brinkman, author of the program, without charge. Brinkman maintains personal control of this program, however, and will not provide source code or allow users to redistribute his program. Other examples of proprietary free-ware on ECMC Linux systems include *ngen*, an alternative, cross-platform Csound score pre-processor developed by ECMC alumnus Mikel Kuehn, and *Vspace*, a sound spatialization program written by Richard Furse.

### The Binary Number System

Computers work with several types of numbers. *Integers*, or "fixed point numbers," are whole numbers, without a written decimal point. 1, 163 and 15427 would be examples. *Floating point* numbers (also called *real* numbers or *floats*) include a fractional part to the right of a decimal point, such as 3.1415, 7.0 or .33.

Computers are constructed from electrical and magnetic components. The simplest (and therefore fastest) types of electrical and magnetic switches useful for representing different values have only two possible states. A computer memory cell may be conducting electricity ("on") or not conducting it ("off"). A magnetic device may be magnetized in one direction, or in the opposite direction. Thus, all of

<sup>4</sup> For more information on the *General Public License* see <http://www.gnu.org/copyleft/gpl.html>

the processing and memory components of a computer system may be thought of as two-position switches. The switching rate (controlled by an internal "clock," and measured by this *clock speed*) is extremely fast, but each switch is in one state or the other whenever a reading is taken.

Digital circuits are not well suited to decimal ("ten-position") representation of integers and floats, but rather to a binary (base two) number system. A single "on-off switch" contains one *bit* (short for **binary digit**) of information, represented by either the character 1 ("on") or by a 0 ("off"). The binary number system is based on the power-of-two series. It takes many bits to represent most numbers. The more bits that are available, the greater the range of quantities that can be represented, or the finer (more precise) the resolution of these quantities.

With sixteen bits, representation of 65,535 discrete integer quantities is possible, as illustrated in the following table. The power-of-two values in this table (for example,  $2^n$ ) should be read as *2 to the nth*, or

$$2^n$$

<i>Power-of-two</i>	<i>Integer equivalent</i>	<i>16 bit representation</i>
$2^0$	1	0000 0000 0000 0001
$2^1$	2	0000 0000 0000 0010
$2^2$	4	0000 0000 0000 0100
$2^3$	8	0000 0000 0000 1000
$2^4$	16	0000 0000 0001 0000
$2^5$	32	0000 0000 0010 0000
$2^6$	64	0000 0000 0100 0000
$2^7$	128	0000 0000 1000 0000
$2^8$	256	0000 0001 0000 0000
$2^9$	512	0000 0010 0000 0000
$2^{10}$	1024 or 1 k	0000 0100 0000 0000
$2^{11}$	2048 or 2 k	0000 1000 0000 0000
$2^{12}$	4096 or 4 k	0001 0000 0000 0000
$2^{13}$	8192 or 8 k	0010 0000 0000 0000
$2^{14}$	16384 or 16 k	0100 0000 0000 0000
$2^{15}$	32768 or 32 k	1000 0000 0000 0000
$2^{15}$	65535 or 64 k	1111 1111 1111 1111

Within the right-hand column (*16 bit representation*) of this table, the *least significant bit* (representing the smallest quantity) is to the far right, while the *most significant bit* (representing the largest quantity) is to the far left. Each bit, from the *least significant* to the *most significant*, represents a successively higher power of two. Note that the least significant bit is  $2^0$  (representing either a one or a zero) rather than  $2^1$ .

Often, as in the digital representation of sound, one bit is used as a sign (positive or negative) bit, resulting in a scale of 16 bit integer values between + 32767 and - 32768. With 24 bits we can represent about 16.8 million discrete values, and integers between 8,388,607 and -8,388,608.

Floating point calculations typically are carried out on 32 bit words (or, on 64 bit systems, on 64 bit words). The bits may be used to represent quantities between 0 and 1.0, or (as with audio samples) between -1.0 and +1.0, or any arbitrary scaling. "Float" (floating point) mathematical operations offer significantly greater resolution (precision), but take longer to execute.

### Bytes and Words

A **byte** is a unit of eight bits. The *bit representations* in the table above each consist of two bytes. A single alpha-numeric (ASCII) character, such as the letter *a* or the digit *3*, is represented and stored as one byte of data.

The basic unit of bits in which a computer works is called a **word**. Word size varies on different computer processors, in powers-of-two, from as few as 8 bits (on some hand-held electronic devices),

through 32 bits (the most common word size today) up to 64 (or, on some "supercomputers," even 128) bits. In general, the more powerful the computer, the larger the word size. All of the ECMC computers employ a 32 bit (4 byte) word size. Registers where arithmetic and logical operations are performed, and memory registers where data is stored, contain 32 bits, which are updated simultaneously.<sup>5</sup>

In your own work, you will be providing data in familiar decimal and integer numbers. These will be translated by compilers and applications into the binary form necessary for execution.

### 1.3. Hardware overview of the ECMC computer systems

The basic hardware components of a computer system include:

- 1) The computer itself, which consists of one or more processor chips, various type of memory, and some type of data communications system or path (usually one or more buss wires), all located on the primary circuit board of the computer called the *motherboard*
- 2) Additional processing chips or circuit boards, usually fitted into backplane slots, that perform specialized functions, such as graphics acceleration, audio or video processing, and network communications
- 3) Mass storage devices, used for permanent or temporary storage of data that is not currently in use, or that will not fit in the computer's RAM memory unit. These devices include
  - disk drives, in various formats, including:
    - ☞ "hard" disk drives, which employ a "fixed" (non-removable) disk with a large storage capacity, measured today in *gigabytes* (billions of bytes, abbreviated *GB*) or, on some multiple-disk high-end systems, in *terabytes* (trillions of bytes, abbreviated *TB*), or even *petabytes*
    - ☞ storage devices that use portable (removable), smaller capacity discs or other types of storage media, such as CD-ROM, CDR and CDRW compact disc drives, DVD ("digital versatile disk") drives, *Zip* drives, "floppy" disk drives and, more recently, *flash* and "pen" drives
  - digital tape drives, which come in various formats, including 4 mm *DAT* data drives, and which today are used almost exclusively on large server systems
- 4) "Peripheral" input/output (I/O) devices, which enable us to communicate with the system, issue commands, receive program output, and perform specialized functions. Such devices include *terminals* (which include a monitor, keyboard and mouse), sound cards or audio converters, MIDI interfaces, printers, scanners, FAX units, projectors, and digital cameras and camcorders.
- 5) Network connections that enable a computer to communicate with other computer systems

The Eastman Computer Music Center includes three studios (rooms 52, 53 and 54) with a cross platform network of Linux, Macintosh and Windows computer systems, each devoted to particular tasks. In previous editions of this *Users' Guide* I provided a tabular listing of the hardware on all of the principal ECMC computer systems, including processor type and speed, hard disk capacity and other "vital statistics." Today, however, we are dealing with commodity hardware. We frequently upgrade the hardware of the ECMC computers, installing more powerful processors, larger disks and new peripheral devices. We sometimes move disks and peripheral devices, such as DVD drives and printers, from one host system to another, and even change the operating system of some of our computers between Linux and Windows as our needs change. As a result, it has become impossible to keep a tabular listing current.

<b>Room 52 : The Linux sound studio</b>
---

This studio is optimized for advanced software-based sound synthesis, signal processing, recording and mixing. The principal computer for these purposes, on which you perform most of your work, is

- A powerful Athlon dual processor *SMP* workstation, running *GNU Linux* software, with the host name

---

<sup>5</sup> Even on a 32 bit system, "double-precision" words of 64 bits (8 bytes) sometimes can be used for the representation of very large and very small integers or floating point numbers. However, such operations run slower on a 32 bit system than on a 64 bit system.

*madking.esm.rochester.edu* and the network IP address 128.151.112.9

This sound studio also contains

- a *Windows XP* workstation named *gesualdo.esm.rochester.edu* with the network IP 128.151.112.005.

This *Windows* machine supplements sound production on *madking*, providing commercial audio software that is unavailable on the *Linux* system. As of this writing the most frequently used applications on *gesualdo* include

- ☞ *discwelder Chrome* : used to author and burn DVD-Audio discs
- ☞ *resample* : an application that converts soundfiles between different sampling rates and bit depths
- ☞ *Vegas* : an application used for creating video/audio productions, and for recording and mixing audio soundfiles; the version of *Vegas* on *gesualdo* also includes authoring software for creating video/audio DVD-V discs

#### **Room 54 : The MIDI studio**

This studio is optimized for real-time MIDI applications, including audio and MIDI sequencing, interactive MIDI applications such as *PD*, and live performances involving hardware and software synthesizers. This studio, too, provides two computers for use:

- a *Macintosh Mac Pro* known as *wozzeck.esm.rochester.edu* with the network IP address 128.151.112.1, used to run *Macintosh* audio software such as *Max/MSP*, *Cubase SX*, *Reaktor* and other *Native Instruments* sampler and synthesis software and *Super Collider 3*
- a *Windows XP* PC with the host name *igor.esm.rochester.edu* and the network IP 128.151.112.3, used to run similar software (e.g. *Cubase SX* and *Reaktor*) on *Windows*.

The *Mac* and *Windows* boxes in room 54 are networked, of course, and often are used in together in the creation of a project. For example, a *Cubase* sequencing project running on one of these systems may include sounds produced by a software synthesis application running on the other system.

#### **Room 53 : The middle studio**

The middle studio is our computer hardware room. The computers used in rooms 52 and 54, along with all of their peripherals except for monitors, keyboards and mice, actually are located remotely in room 53 in order to minimize noise in the two sound studios. Room 53 also contains several additional computer systems and peripherals, and thus can be a rather noisy place. *ECMC* users employ the auxiliary computers in the middle studio for general purpose and utility computing functions, such as editing text files and documents (like this one), making copies of *CD* and *DVD* discs, reading e-mail and newsgroups, using internet resources such as web browsers and *ssh*, backing up data, editing soundfiles or working on audio applications while someone else is using room 52 or 54, and reading printed documents and the *ECMC* collection of books and journals (which must never leave the studios). As of this writing this room contains the following computer systems:

- a *Linux* workstation with the host name *sound.esm.rochester.edu* and IP number 128.151.112.7.
- a *Windows XP* workstation with the host name *fury.esm.rochester.edu* and IP number 128.151.112.003.

You can use these two systems for burning and copying compact discs, for web browsing, reading your email and similar tasks not directly related to your compositional work in the studios.

By the time you read this, one of these two machines may have been replaced by a *Mac G4* formerly used in the *MIDI* studio.

- a *Linux* server with the host name *lulu.ecmc.rochester.edu*. This system is the *ECMC* web server, handling requests from anywhere in the world for access to the *ECMC* web pages. You will not have login access to this system.
- a *Linux development* system, used by staff members for testing and debugging new or updated software before it is installed on *madking*, and for maintaining the *ECMC Turnkey* software package
- a *Linux* system that is used exclusively as sound server for the playback of 8 channel soundfiles in room 52. You will not have direct login access to this machine, which is discussed more fully within section 3.

Please note that you should **never tie up either the MIDI or Linux sound studios** (rooms 54 and 52) **with non-music related tasks** such as web browsing or reading email. Always use the machines in room 53 for such purposes. Additionally, you should not try to reconfigure any ECMC system, nor, without permission from a staff member, install any software or download large files. If you wish to use machine *sound* or machine *fury* to make a temporary small download, copy this download immediately to some removable media (e.g. to a cd, dvd or flash drive) and then delete it from the ECMC system.

Two additional ECMC laptop computers, a Dell model 8200 with Windows XP and Linux installed, and a Macintosh PowerBook, also are available, but for security reasons they are not kept on line and are locked away in a cabinet. These portable systems are used for remote recording and play-back functions involving ECMC users.

Room 53 also contains a wireless network switch and access point (discussed later) and a networked *HP 8000N* printer. Files and documents created on all ECMC computers (except the laptop) with page sizes up to 11x17 inches can be printed out on the heavy duty *HP* printer.

Our Linux and Macintosh systems function not only as client machines on the internet, but also as servers, and assuring network security for these systems is among our paramount concerns and problems. The ECMC Windows systems, by contrast, do not have server software installed, and can function only as client systems on the internet. However, because viruses, trojans and other malware are so prevalent on Windows systems, security concerns are critically important on ECMC Windows machines.

Owing to some limitations in the user manager software of our Windows and Macintosh systems you generally should not try to log off or log on to the ECMC Windows and Mac machines. Rather, we keep a single, generic user account named *ecmc* active almost all the time on these systems. However, users should save documents to their own individual folders. For convenience, we currently keep a similar generic user account running continuously on Linux machine *sound*, and you need not log on nor log off this machine. On the Linux system *madking*, by contrast, you must log in to your own account before you can access the system and you must log off when you are done. Failure to do so will leave your files accessible to anyone, and also could compromise the security of these systems.

Students taking the *Advanced Computer Music* class (CMP 421-2) generally will be given accounts on the two systems in room 52 (*madking* and *gesualdo*), and also, if requested, on the two MIDI studio computer systems (*wozbeck* and *igor*), and also are free to use *sound* and *fury* in room 53 for routine computing tasks.

=====

#### 1.4. A more detailed look at computer hardware

##### The Computer : CPU, memory and data I/O system

A computer *motherboard* holds three primary components:

- 1) one or more central processing and control units (abbreviated *CPUs*)
- 2) memory registers, where instructions and data currently in use are stored
- 3) *buss* wires, along which data is transferred between the CPU , RAM, cache memory and other devices

In addition to the *CPU* itself, additional chips<sup>6</sup> or circuit-board cards connected to a buss often handle specialized functions, such as video and audio processing, network connections and controllers for disk drives and other devices.

The *CPU* controls the operation of all of the hardware of a system, much as the human brain regulates all of our body systems. Among the primary characteristics of a computer are its speed, its memory and word sizes, and the speed of its I/O (input/output) data transfers with other devices. These determine the execution time for programs (including which types of operations can be performed in real time), as well as the complexity of operations that can be performed, and the number of simultaneous processes that can be supported before the system slows to a crawl. Audio signal processing is among the more

<sup>6</sup> "Chips" are integrated circuits — electrical components, etched onto a small piece of silicon, designed to perform specific types of operations.

taxing jobs that a computer can be asked to perform, often requiring millions of computations to create a few seconds of sound, or large amounts of data transfer for the playing, recording, processing and mixing of sounds. Video applications such as image processing place even greater demands upon a computer system.

The CPU performs integer or floating point mathematical operations required by programs. Integer arithmetic is much faster for computers (as it is for people) than floating point calculations. The speed of a computer system can be measured in several ways. One preliminary — but sometimes misleading — indication is the raw clock speed of the CPU, measured in gigahertz (GHz) or (on older machines) in megahertz (MHz). Today, clock speeds of between about 1 and 3.6 GHz are common. However, the total *throughput* (time required to complete a task) of a system is also affected by many other factors, including

- the efficiency of machine cycle operations,<sup>7</sup> and of the operating system
- the speed of a computer's I/O *buss* interfaces, through which data is transferred between the CPU, RAM and various devices; on machines with relatively slow buses, a CPU may spend much of its time cycling idly, waiting for data to be sent to or received from other devices
- the speed and transfer rates of the disks and peripheral devices themselves
- optimizations built into a system for particular tasks.

As noted earlier, *Symmetrical multiple processor (SMP)* systems, currently ranging from motherboards with two, four or eight CPUs on desktop systems up to banks of 128 or more processors on some "enterprise" class servers, are becoming increasingly common. *madking* and *wozzeck*, the *Mac Pro* in the MIDI studio, have two CPUs. Currently, all of the other ECMC systems employ a single CPU.

Computers with two or more processors will provide noticeably better performance than a comparable single processor system when two or more programs are run simultaneously. However, when a single large application such as *Cubase*, *Ardour* or *Pro Tools* is being used, the additional processor(s) will provide only a marginal performance boost unless the application has been specifically written and configured to take advantage of available parallel processing. Most audio applications do not yet make efficient use of multiple processors.

Part of being a smart power user is knowing the hardware and software capabilities of the system you are using. On an SMP machine such as *madking*, you generally can run two processor-intensive jobs (such as sound synthesis and audio mixing) simultaneously. On a slower single-processor machine it usually will be more efficient to run these two jobs in succession.

When we issue a command to a computer to run a particular program, the code (instructions) for the program is read into the computer's memory from a disk file. When all of the primary instructions and preliminary data necessary to run the program have been entered into memory, the program is ready for execution or compilation. One by one, the instructions are transferred from memory to the CPU and executed. The results are then written back into the memory unit. *Interactive* applications, such as word processors, soundfile editors and MIDI sequencers, allow us to issue instructions and to enter or alter data while the application is running.

Only user programs and system processes, and their data, that are currently being run are stored in memory. The memory unit of a computer is often referred to as *RAM* ("random access memory"), because the CPU can access any memory register as quickly as any other register. RAM sizes today vary between 512 MB and several terabytes on large servers. In addition to RAM, most computer systems today include a small *primary cache* memory, and one or two larger 512 KB to 4 MB *level2 (L2)* cache memory chips, which store frequently or repetitively accessed instructions within a program, or recently accessed data, or data likely to be accessed soon. Caching operations provide provide a much greater performance increase for some programs than for others. In addition to RAM and cache, *VRAM* (video RAM) memory chips ranging from 8 to 64 MB are employed on graphics cards and video adapters to accelerate graphical displays and video processing operations).

Computer memory (RAM) has become relatively cheap during the past few years, and all of the ECMC music production systems have at least one gigabyte (1 GB) of RAM. At the same time,

---

<sup>7</sup> for example, some CPU architectures can accomplish somewhat "more work" in a given number of machine cycles than other processor designs

however, many operating systems and graphical software applications have grown much larger, sometimes with inefficient, poorly written code and with ever more features and options (some of which are rarely used) — a phenomenon called "*bloatware*". Larger, "feature-rich" applications require more memory as well as more processing power. As a result, it is rare — although by no means impossible — for ECMC users to exceed that RAM capacity of a system.

All types of computer memory are volatile. If a system "crashes," either from a hardware or software malfunction or from a loss of power, everything in memory — even the operating system — is lost, and must be re-loaded from disk or cd-rom or by some other method. This is called *booting* (or, after a crash, *rebooting*) a system.

### Data transfer formats

The operation of each peripheral device in a computer system (such as a hard disk, printer or audio card) is controlled by a *device driver* program and in some cases also by a circuit board *controller*. Devices come in various *formats*, which determine how data is transferred between the device and the computer. Common formats today, in approximate order from slowest to fastest, include:

- *IDE* : old and slow; no longer used
- *EIDE* : a faster, "enhanced" *IDE* format, used on most PC systems, but now gradually being replaced by *SATA* drives
- *SCSI* ("*Small Computer Systems Interface*") formats, which typically provide better performance and noticeably better reliability (usually carrying a three year warranty as opposed to the one year warranty now standard on *EIDE* disks). However, *SCSI* disks have smaller capacities and are more expensive, and thus are used on servers more often than on workstations.
- *ATA* and, more recently, *SATA* (*Serial ATA*) and *SATA2*; high quality *SATA* disk drives offer performance that approaches or equals that of high quality *SCSI* disks, but at less cost. *SATA2* disk drives have replaced the preceding disk formats on many new workstation systems.

Two newer data transfer formats, introduced in 1998, include *USB* and *IEEE-1394* (also called "*FireWire*", especially on Mac systems). The original *USB* format was comparatively slow but inexpensive and very easy to install and use. *USB* devices can be "hot swapped" (connected to, or disconnected from, a computer system while the computer is up and running). *USB2* offers significantly enhanced bandwidth (input/output speed), but in our experience remains problematic for some audio interfaces. "*FireWire*" format combines similar convenience with much greater speed, but is more expensive than *USB*.

### Storage devices: internal, external and removable disks, USB flash devices, CD and DVD discs

A disk or tape device can be mounted either *internally*, within a drive bay within the case of the computer, or else *externally*, in a metal box connected by a cable to a port on the computer. One or more "*hard*" disk drives form the primary bulk storage devices on most computer systems. A "*hard*" (non-removable) disk drive is a "turntable" that spins a three and a half inch magnetically coating disk rapidly at speeds ranging today between 5400, 7200, 10,000 and 15,000 rpm. While the disk is spinning, movable read-write heads read and/or write data to and from the disk. Data from any point on a disk can be located, accessed and read approximately as quickly as data from any other point. Thus disks, like RAM memory, are "random access" devices, as opposed to "sequential access" devices like a tape drive. Besides its rotational speed, important characteristics of a disk drive include its

- storage capacity: generally between 40 GB and about 1 terabyte (1000 GB) today
- *seek time*: how long, on average, it take the drive to access a particular location on a disk; seek times of between five and nine milliseconds are common on high quality drives today
- *transfer rate*: how much data can be read from or written to the disk in a second; current formats theoretically can achieve transfer rates up to 80 megabits or more per second, although actual throughput rates always are somewhat lower.
- reliability
- noise; some drives "chatter" a lot when reading or writing data and "hum" or whine while spinning — highly undesirable characteristics within an audio environment

Seek times and transfer rates become particularly important in applications such as real-time audio mixing of soundfiles, where the disk drive must read simultaneously from several soundfiles scattered at

various points on the disk. Disk drives with rotational speeds of 5400 rpm can read (playback) or write (record) only three or four audio tracks simultaneously, which is not sufficient for professional audio use. Disks rotating at 7200 rpm generally can handle 12 or more audio channels simultaneously.

When a disk fills up to 90 % or more of its capacity, files — especially large files, like soundfiles — become highly fragmented, split up into small chunks at various locations on the disk. When this happens, system performance begins to suffer — jobs that read from or write to the disk take longer to execute, and real-time audio applications may include hiccups, clicks and other garbage within the sound when data is not received in time.

The hard disks of Windows and Macintosh systems should be defragmented periodically with a defragmentation program. The disk management and journaling file systems such as *ext3* and *ReiserFS* of Linux computers accomplish disk defragmentation continuously and automatically, so it is not necessary to perform this task manually.

Large capacity portable external USB2 and IEEE-1394 disk drives that can be easily connected to and then disconnected from a running computer (without the need to reboot) are very useful for backing up larger files and entire projects and for accessing occasionally-used data not permanently stored on the system's internal hard drives.

Most computer systems today are configured to *automount* USB and Firewire devices within a few seconds, automatically *mounting* (connecting) the data on the device to the filesystem for immediate use. When the device is disconnected or powered down the system then will automatically *unmount* it. Such hot plugging is very convenient for the user (who does not need to do anything before or after using the device), but it does increase system overhead and degrade system performance somewhat because the system must continually poll (monitor) the status of the external ports. Sometimes system administrators will choose to turn off *autopolling* of USB, IEEE-1394, cd, dvd and other ports and devices in cases where system performance is critical, as in the live performance of a computer piece, or if a slower system is being used for computationally intensive tasks. In such cases, users must manually mount the device before it can be used, then manually unmount it after use to avoid possible file system corruption.

**Removable storage devices:** Older types of removable disk drives, which enable users to insert and eject portable disks into a drive, offer less capacity and speed than fixed disk drives. Small three and a half inch "floppy" disks (which for more than twenty years have not been "floppy," or "bendable") can store only a small amount of data (usually 1.4 Mbytes), and for this reason are rarely used today. For several years the Iomega company's 100, 250 and 750 MB model *Zip* drives were the dominant type of removal disk drive on personal computer systems. Today, however, the cheap cost of cd and dvd disc drives and media, and of USB *flash* drives and swappable external hard disk drives, have made removable disk drives such as the *Zip* and floppy disk pretty much obsolete.

USB **flash** drives are small, lightweight, inexpensive permanent storage devices that currently can store between 32 MB and 64 GB of data. Capacities between 1 GB and 4 GB are most common. *Flash* drives generally are hot swappable, and can be connected to and disconnected from a USB port on the computer case while the computer is running. If USB automounting is enabled on the system, the drive will be attached to the file system automatically within a few seconds and become available for use. (If automounting is not enabled, the user will need to mount the flash stick before use and then unmount it after use.) *Flash* drives, covered in more detail within Section 5, are an excellent way to back up a day's work and to copy and transfer data between computer systems, and I recommend that you purchase a flash drive for use on ECMC systems. Note, however, that because of their relatively modest storage capacity *flash* drives are not suitable for archiving many large soundfiles.

**CD and DVD drives:** cd-rom disc drives (note the spelling here: "*disc*," not "*disk*") are similar in design to audio cd players, except that the discs most often store standard computer ASCII and binary file data rather than audio samples. The "*rom*" in the device name indicates that the discs are "**read only media**": users can read data from the discs, but not write to them. Thus, cd-rom discs are used today primarily for the distribution of commercial software. However, with appropriate software applications cd-rom drives also can be used to play audio cds through the computer's audio system. Standard compact discs have a capacity of about 650 MB of data (800 MB capacity discs also are available) or about 74 minutes of 44.1k stereo audio.

*cdr* "burners," available on most of the ECMC systems, are drives that enable us to write (create) data on cd discs as well as to read (or play) these discs. *cdr* discs can be written only once, and usually only in one continuous pass. Data on the disc cannot be erased, altered or (usually) appended, and if something goes wrong during a recording the disc is worthless. The cost of quality *cdr* media has dropped to less fifty cents per disc when purchased in spindles of 25, 50 or 100 discs. However, beware of the uneven quality of some super-cheap no-name CD and DVD discs.

*Re-writable* cd drives and discs, called *cdrw*, allow one to overwrite the data on a disc (to reuse the disc). All recently manufactured cd burners can read and write to both *cdr* and *cdrw* discs.

*DVD* drives, introduced in 1999, read from *dvd* ("digital versatile disc") platters, which are the same size as compact discs but store much more data. For some strange reason, the capacities of DVD discs are measured in *G bytes* (1000 bytes) rather than *gigabytes* (*GB*, or 1024 bytes). Thus a *G byte* equals .977 *GB*.

Currently, most DVD discs have a capacity of 4.7 G bytes (equal to 4.58 GB), more than six times the capacity of a cd). Double layer 8.5 G byte DVD discs also are now available, and DVD capacity eventually is scheduled to increase to 17 G bytes. The laser pits of a DVD disc are much smaller and packed more closely together than the pits on a cd. DVD drives can read cd discs, but cd drives cannot read the more densely packed dvd discs.

There are several types of DVD formats, including:

- *DVD-ROM* : Like *cd-rom* drives, *dvd-rom* drives can only read discs, not write to them. On computer systems these drives have largely been superseded by *-R* and *+R* writable format drives.
- *DVD -R/-RW* : Like *cdr/rw* drives, *dvd-r* format allows users to write either to permanent (*-r*) or to erasable, re-usable (*-rw*) discs. This is the most common dvd format today. *dvd-r* video/audio discs can be read by most standalone dvd decks used within professional and home entertainment systems, and this is the only format that should be used when creating *DVD-A* (audio) discs.
- *DVD +R/+RW* : This is an alternative format to *-R* that offers some performance advantages — in particular, faster writing. For this reason *+R* and *+RW* formats have become the most common formats for data backup, but not for video and audio DVDs, where *-R* format still provides greater compatibility with all types of DVD playback drives.

Most dvd drives manufactured since 2003 can read and write dvd discs in both *-R/-RW* and in *+R/+RW* formats, as well as *cdr* and *cdrw* discs. More specific information on using *dvd/cd* drives in the ECMC studios is included within section 5 of this *Users' Guide*.

## Video Monitors

Video I/O devices for interactive use with a computer have traditionally been called *terminals*, and today include a video display monitor, a keyboard and a mouse (or a touchpad, trackball or similar "point-and-click" selection device). During the 1980s and early to mid 1990s ECMC Unix systems usually included two or three terminals, and two or three users often logged on simultaneously to these systems. For the past decade, however, each of our computers has included a single terminal, although it is possible to log on remotely to most of our systems from other computers.

For many years the most common type of video displays were CRT (cathode ray tube) monitors, which come in sizes between 14 and 30 inches (measured diagonally, although the actual viewing size always is slightly smaller). Laptop and notebook computers typically contain smaller, flatter (and often somewhat dimmer) LCD (liquid crystal diode) monitors. Flat panel LCD monitors are rapidly replacing CRT monitors on desktop systems as well. Although still slightly more expensive than conventional CRT monitors, flat panel models introduce less visual distortion around the edges, take up less room, are lighter and more portable and are less likely to induce noise into nearby audio circuits. For these reasons, all of the music production systems in the ECMC studios employ flat panel LCD monitors, usually at a resolution of 1280x1024 pixels. (A *pixel* is one dot on the monitor display.) Some high resolution monitors, used mostly for video and CAD (computer-assisted design) work can provide displays up to 1600x1280 resolution or even higher.

## Sound cards and audio interfaces

Sounds are recorded and stored as binary data files. The data contains thousands or millions of numbers. We cannot hear these numbers directly, of course. All digital audio playback applications and components require a *digital-to-analog converter (DAC)*, which converts the number stream into a corresponding electrical (analog) voltage signal that is sent to a power amplifier and loudspeaker. An *analog-to-digital converter (ADC)* performs an opposite function, taking a reading of the voltage level of an audio signal at regular, very short time intervals, and producing a corresponding stream of numbers.

The motherboards of many computers include an on board sound chip that performs digital to analog conversion, but the audio quality of such chips is not adequate for professional quality audio work. Sound cards fitted into a PCI, USB, Firewire or other slot in the computer backplane can provide better audio quality, but sometimes suffer from noise induced by switching transients from the motherboard and other cards within the computer enclosure. External converters and audio interfaces that include converters, situated outside the computer and usually connected to a signal routing card within the computer backplane, generally provide a quieter environment for audio processing and provide better cabling options. The ECMC music production systems employ a mix of high quality soundcards and companion external interface boxes manufactured by the German RME corporation and *Delta 1010* models manufactured by the M-Audio company.

### 1.5. Computer Networks

Until the early 1970s almost all computing was done on large mainframe computer systems, often with twenty or more terminals. Users had access to a large data base. However, during periods of heavy use (which often extended through most of the twenty-four hours), such systems sometimes slowed to a crawl servicing so many users and processes. During the mid 1970s, smaller, less expensive *mini-computer* systems were widely used. Typically, a mini-computer system could service between two and five simultaneous users, and many were dedicated to particular tasks (such as making music). The development of still smaller and cheaper *microcomputers* in the years around 1980 led to the advent of inexpensive single-user personal computing systems, such as the early IBM PC (running early versions of Microsoft's *DOS* and then *Windows* on Intel processors) and Apple Macintosh, as well as microcomputer systems manufactured by Atari, Amiga, Commodore, NeXT and other hardware vendors.

During the mid 1980s attempts to combine the advantages of mainframe computing (access to large data bases and storage capacity), with those of personal computers (fast response time and convenience for individual users) led to the development of *distributed computing systems* over computer networks. In a *local area network (LAN)*, one computer may act as a *file server*, or *host*, servicing requests from several *client* workstation computers for access to disks, printers and other peripheral devices, or else for files, programs and other types of services. Freed from the need to control all of these devices and services, each client CPU can give almost undivided attention to user processes. Groups of LANs, in turn, can be connected into an *Extended Local Area network (ELAN)*, such as the University of Rochester network, which then are connected to the *internet*, a world wide "network of networks."

The ECMC systems are connected by networking hardware and software, comprising a LAN with the internet domain name *esm.rochester.edu*. However, there currently is no file server machine on our LAN. Rather, each of our computers is self-sufficient, within its own hard disks and complete, self-contained file system. All ECMC computers also are connected by a very fast local 1 Gbit Ethernet system, which in turn connects to a slower 100 base T Ethernet system that services all of Eastman, then by routers and fiber optic cabling to the U. of R. ELAN, and, via U. of R. gateway machines, to the internet. We use U. of R. servers as gateways to access web and email sites, and for a variety of other network functions.

ECMC Linux system *lulu* functions as our network web server, not only for our own users, but also for users across the internet who wish to access the ECMC web pages. Using this same networking hardware and software, ECMC users can log on to any of our computers from any other ECMC computer (for example, logging on to their Linux *madking* directories while working on Windows machine *fury* in room 53 or the Mac system *wozzeck* in the MIDI studio). Additionally, using the same *ssh*-based applications, you can log on our Linux and Mac systems remotely, from any computer connected to the internet that has *ssh* software, and also copy files back and forth between local and remote systems.

*Wireless networks*, which transmit data by microwave transmission through the air (the same basic technology as cellular phones) are becoming increasingly common because they eliminate cables. The ECMC studios include a D Link *five port wireless switch* and a companion D Link *access point* in room 53 that we use to connect some of our computers to the ECMC LAN. ECMC users are welcome to bring their personal laptop or notebook computers into the studios and connect them to the ECMC LAN. Before you can do this, however, you must have appropriate wireless networking hardware and software installed on your computer, and for security and practical reasons you must register some information about your computer with the Eastman Technology and Music Production (TMP) technicians who support wireless access to various computer systems within the School. Call one of the TMP desktop support consultants at 274-1160 or 274-1161 to register your computer, and see an ECMC staff member to obtain the appropriate ECMC WEP key number to use with the wireless access software on your laptop.

### 1.6. Software (direct) synthesis and MIDI applications in the ECMC studios

The establishment of *MIDI (Musical Instrument Digital Interface)* communications protocol standards in 1983 made it possible for general purpose microcomputers to control and store performance data that drives digital synthesizers, samplers and other audio processors. Between the mid 1980s and the mid 1990s the Macintosh platform provided the largest base of quality MIDI software, but for more than a decade MIDI software of comparable quality has been available for *Windows* systems. Today, high quality open source MIDI software for GNU Linux systems is also available and under rapid development.

Besides controlling MIDI devices, a computer itself can run software programs that calculate the numbers that represent sounds or sound processing procedures. *Software synthesis* (sometimes called *direct synthesis*) is slower than MIDI control of digital audio hardware, and requires greater knowledge of programming techniques. However, since all of the audio processing procedures exist in software, they offer greater flexibility and extensibility, and many more possibilities than the fixed hardware architecture of a particular synthesizer or effects processor. Hard disk-based audio workstations can mix, manipulate and transform sounds in a variety of ways, replacing tape recorders, mixing consoles, effects processors and other audio gear used in traditional recording studios.

Sophisticated audio synthesis and signal processing was developed initially (beginning in the late 1960s) on Unix systems at academic and research computer music centers (the only institutions that could provide access to the expensive digital hardware of the time). The music software base on our Linux systems reflects this legacy as well as its continuing evolution, which now generally takes the more egalitarian form of open source software development, distribution and sharing by individuals and teams of programmers and musicians from throughout the world.

Since the mid 1990s powerful commercial music software that performs similar tasks has been available for Macintosh and Windows users. There are advantages and disadvantages to commercial software and to open source software. If you become thoroughly conversant with both of these software streams you can make informed choices as to which programs and applications are best suited to your working preferences and to the realization of particular musical objectives, and will have more resources from which to draw in your compositional work.

#### *A brief history of the ECMC studios*<sup>8</sup>

*Room 52:* The Eastman Computer Music Center was established in 1980 with the installation of a DEC PDP-11 computer system dedicated to music production. This system, which initially included 132 **k**bytes (sic!) of RAM and an 80 MB disk (both respectable capacities for the day) often crashed several times a day, but included very high quality DAC and ADC converters and was used to create some of the ECMC music software and *sflib* soundfiles (such as the tam tam in */sflib/perc*) that we still use today.

During the mid 1980s the PDP-11 was replaced by a network of *Sun3* Unix workstations, and by 1990 software-based recording, sound synthesis, signal processing and mixing operations were performed on NeXT Unix workstations. From the mid 1990s through 2002 SGI Unix workstations were employed for these operations. Our first Linux box was installed in 1998, and by 2003 Linux music

---

<sup>8</sup> A more detailed, if rambling, memoir on the history of the ECMC studios is available on the ECMC web site.

software had become sufficiently sophisticated to enable us to retire the last of our aging SGI machines.

*Room 54:* In 1983 we installed a *Mac Plus* computer to augment our existing analog synthesizers and electronic music gear and to make MIDI capabilities available to ECMC users. During the late 1980s this primitive machine was replaced by a Mac 8100, and in the early 1990s by a Motorola *Star Max* Macintosh clone. During the later 1990s we experienced increasing incompatibility and instability problems with our Mac software (particularly with *Pro Tools*, which did not get along with the rest of our installed audio software applications), along with dissatisfaction with the high cost and mediocre performance of Mac hardware of this period and so migrated our MIDI studio computer operations to a Windows system. By the spring of 2003, with the availability of OS X-based audio software for the Mac, these instability and incompatibility problems on the Mac platform had largely disappeared, and we installed a Power Mac G4 (replaced with a *Mac Pro* in 2007) in the MIDI studio to complement our existing Windows system.

Each of the ECMC computer systems in rooms 52 and 54 provide a wide range of both MIDI and audio signal processing capabilities. However, our MIDI studio is optimized for real-time and MIDI-related tasks; we try to provide the very best, most powerful Macintosh and Windows MIDI and audio software in this studio, though not necessarily what one finds in a commercial recording studio designed primarily for pop and commercial music and voiceover production. Similarly, the Linux and Windows PC systems in room 52 are optimized to create the best environment we can provide for sophisticated audio signal processing and synthesis. ECMC users are encouraged to become familiar with the resources of all of our computer systems, so that they can employ the most powerful resources of the Center for a variety of specialized musical tasks. In reality, there is only one ECMC studio, which encompasses an integrated network of Linux, Macintosh and Windows computers, software and audio peripherals spread through three rooms.

### 1.7. Crashes, hangs and rebooting

Computer "crashes" and "hangs" can result either from hardware malfunctions, but much more often are caused by software problems or by user error. When a computer *hangs*, it becomes trapped in an endless loop or an impossible operation (such as trying to access a non-existent memory location, known as a *segmentation fault*) somewhere in the middle of a process. The operating system is unable to terminate the process or to regain control of the system. With multitasking operating systems usually only the process itself is affected: its window(s) will freeze, and will not respond to any input. In such cases, it generally is possible to kill the errant process (and its subprocesses) from another window, and then to resume work. Occasionally, however, a hung process will paralyze the entire system, and it will be necessary to reboot the system.

While a *crash* most often results from user error, system overload or from a badly written program, it may instead portend a more serious system hardware or software problem, such as a bad sector on a disk or a failing component on a circuit board. The operating system knows that something is wrong, and shuts down to prevent corruption of, or further damage to, the file system. All current operations are aborted, wiping out everything in RAM. During rebooting, the operating system is read back into memory from a disk, the file system is checked to see if anything is missing or duplicated, and most problems are corrected automatically. Our Linux systems employ *journaling* file systems, in which all changes in the file system are logged almost instantly. As a result, file system checking and repairs during a reboot generally are accomplished very quickly with little if any loss or corruption.

*Windows XP* systems employ less robust file system updating procedures, and upon rebooting after a serious crash they may require check-and-repair operations lasting several minutes. Occasionally — although much less frequently than in the past — a *Windows* reboot after a crash will result in the feared *blue screen of death* — an impenetrable error list of numbers highlighted on a blue background. Usually, rebooting again (and then, perhaps, *again*) eventually will clear the problem. If not, extensive system rebuilding may be necessary. Always advise a staff member if you are confronted by the *blue screen*. Macintosh systems employ a proprietary file system called *HSF* + that is generally robust but that now and again may require simple or extensive repairs after a crash.

If one of the ECMC computers appears to lock up on you, you should only perform a "hard restart," by pushing the *stop* or *restart* button on the computer case, as a last resort. Simply "pulling the

plug" is never a "good" solution, since it may result in corruption of the file system. Reboots made from software are much better than simply hitting the hardware shutdown/reboot button. First try quitting the process on which you are currently working, or accessing another window with the mouse. Is it really the entire computer system that is locked up, or only the the currently active process or the monitor screen? Make sure that the screen saver has not kicked in (tap the mouse button or the *enter* key) and that you have not inadvertently issued a *suspend* or *lock screen* command or (on Windows and Macintosh systems) put the computer into *sleep* (or "hibernate") mode. If a staff member is available, get help. If all else fails, halt and restart the computer with the hardware *reset* button, and report the crash or hang to a staff member.

 **Procedures for restarting Linux systems:**

- Try clicking in, or opening, a shell window. If this succeeds, type

*ps aux*

in the terminal window, note the process ID number(s) of the problematic job, and kill the rogue process:

*killall process\_name* (where *process\_name* is the name of the hung application)

or else

*kill-9 process\_number*

(The *killall* command is generally preferred, because it will terminate all subprocesses launched by the main application in addition to the application itself.)

Example: If the application *audacity* hangs, a *ps aux* command might display a line such as

```
allan 4486 1.0 1.0 65324 10936 pts/2 S 07:45 0:00 /usr/bin/audacity test.wav
```

I can kill this job by typing either *killall audacity* or else *kill -9 4486*

Alternatively, type *xkill* in a shell window. The cursor will change to a skull and crossbones. Move the cursor to the window of a application you wish to kill and then click the left mouse button.

- If this doesn't work, click on *System* on the task bar and try to log out or, better (to force a file system check), click on *Shutdown* and then on *Restart*. If you believe that there is a problem with the system that will not be corrected by rebooting, click shut down the system instead.
- If you still get no response you will have to bite the bullet and do a hardware restart. Press the *reset* button on the computer case. After shutting down the system, wait a few seconds, then push this button again to initiate rebooting. While rebooting, Linux may inform you that some disk partitions were not mounted "cleanly," and, if so, will run system checks and repairs if necessary on the file system.

 **Procedures for restarting Windows systems:**

- If the system does not respond to any mouse clicks or movement, hold down the **Ctrl**, **Alt** and **Delete** keys simultaneously. On *Windows XP* systems this should open a *Task Manager* window.
- In the *Task Manager* window, select the process that is hung and click on *End Task*. You may need to kill more than one process.
- If this doesn't work, repeat the *Ctrl/Alt/Delete* command, if necessary, then select *Shutdown and Restart*.
- If this still doesn't work, press the power button on the computer case. After shutting down the system, wait a few seconds, then push this button again to initiate rebooting.

☞ **Procedures for restarting Mac OSX systems:**

- To terminate an errant application, you can use the *ps aux* and *kill -9* Unix commands described under the boxed *Procedures for restarting Linux systems* above work on Mac OSX as well.
- Alternatively, if the mouse still works, you can attempt to *force-quit* the application. However, this may cause the Mac to crash, so do not perform this procedure lightly. Hold down *Option-Apple-Escape*, select the errant program from the list, then select *Force quit*
- If you are unable to terminate the program, select *Log out* or *Restart* or (if you believe that the system is not in a good state) *Shutdown* under the *File* menu.

(After a nasty crash it may be necessary to run the *Disk Utility* or *fsck* to repair the file system. For some repair tips, see <http://www.jmu.edu/computing/mac/crashesOSX.shtml>)

### 1.8. Studio security and usage policies

Occasionally demand for studio time and/or soundfile disk space exceed capacity. The following procedures have been established to enable us to work together amicably rather than crawl all over each other like a swarm of angry insects. Violations of studio usage regulations or courtesies will result, initially, in a nasty e-mail message, then may lead to a loss in studio privileges. Neglect of studio security policies will be dealt with very severely, even upon a first infraction.

#### Security regulations

Every so often students not authorized to use the ECMC studios, or individuals with no connection to Eastman or the U. of R., attempt to convince students to allow them into the studios.

**NEVER ALLOW ANYONE WHOM YOU DO NOT RECOGNIZE AS AN AUTHORIZED STUDIO USER TO ENTER THE OUTER DOOR TO THE STUDIOS.**

**NEVER DEMAND TO BE ADMITTED TO THE STUDIOS BY SOMEONE WHO DOES NOT RECOGNIZE YOU.**

If someone gives you a hard time about this policy, immediately contact a staff member or, if necessary, ESM security (extension x13, or 5333).

**STUDIO EQUIPMENT AND SUPPLIES, INCLUDING HEADPHONES AND MANUALS, MUST NEVER BE TAKEN FROM THE STUDIOS FOR ANY REASON.**

**MICROPHONES, MIC STANDS AND OTHER EQUIPMENT SHOULD NEVER BE MOVED BETWEEN THE STUDIOS unless you first consult a staff member and make certain to return them immediately after use.**

If you wish to borrow any studio equipment, materials or documentation briefly for some exceptionally good reason, consult with a staff member well in advance.

In particular, we have had ongoing problems with headphones disappearing from the studios. This is especially disturbing because it means that we have been ripped off by a few of our own users. For this reason we no longer keep our higher quality headphones easily accessible within the studios, but instead provide them only on a sign-out basis, and all of our users suffer the inconvenience of this policy.

When you first enter any of the three studios take a careful look around to determine if any studio equipment is missing or broken, and if so report this immediately by email to a staff member. If you break something, report it in the same way.

Violation of any of the policies above will result in loss of access to the ECMC studios. Any incidence of stealing (or of unauthorized "borrowing," even "just for a very short time") of studio equipment additionally will result in immediate and permanent loss of studio access and will be reported to the ESM Dean of Students office. Put simply, we need to have absolute trust in the integrity of all of our studio users.

#### *Accessing the studios and reserving studio time*

Access to the ECMC studios for authorized student users is provided by means of an ID card reader next to the entrance to the studios. Your student ID number must be keyed into the reader or the

door will not be unlocked. After entering the anteroom, you can use the keys near the doors to rooms 52 and 54 to unlock the doors to these studios. To unlock the key (in order to unlock the door), set the three digits of the key to the current combination and unloop the key so it will reach the door lock. After unlocking the door, loop the key back around its lock (so that it will not reach the door) and scramble the three digit combination of the key lock.

Each Thursday sign-up sheets for the following week are posted on the bulletin outside the entrance to the studios. Depending upon demand for studio time, time quotas may or may not be included on the MIDI studio and Linux sound room sign-up sheets. If you wish to reserve permanent working hours, consult the studio director.

After 6:00 pm, a user may sign out additional hours, beyond any quota restrictions, for remaining free time on the following day. Any time that either of our two studios is not being used, it may be signed out on an ad hoc basis. However, the studios must be signed out whenever they are in use.

If you are unable to come in to work during a time you have signed out, it is your responsibility to erase your name or to call the studios (274-1578) and ask someone to do this for you. Any user who consistently signs out time then does not show up may face loss of studio privileges.

When your assigned studio time is up and another user is ready to use the room, be sure to terminate all of your processes (especially *jack* on Linux systems), including any background jobs.

All studio users are subject to the Eastman School building hours. The building is generally open daily between 7:30 am and 11:00 pm whenever classes are in session, but building hours are more restricted during vacation, holiday and certain summer periods.

The outer door to the Computer Music Center must be kept locked at all times to minimize the chances of theft. **The doors to rooms 54 and 52 also should be kept locked when not in use, especially during evening and week-end hours.** (Room 53 should not be locked.)

### **Obtaining supplies and materials**

The Computer Music Center purchases removable discs, tapes and other supplies in bulk at wholesale prices and sells them at cost to all users. The current prices for available studio supplies are posted on the wall outside room 53. Pay the cashier on the first floor the correct amount for the supplies you want, ask her or him to credit the computer music account (2-11196) and obtain a receipt. Bring this receipt to a staff member to obtain the supplies.