Electronic Musician

AN ACT III PUBLICATION
AUGUST 1989 VOL. 5, NO. 8

features

The Pro Studio Comes Home
Thanks to DAT, MIDI, and computers, you can record CD-quality projects at home—on a very reasonable budget. **32**
by Craig Anderton

CD Packaging: State of the Arts
Get hip to the best ways of packaging your CD to reach your market. **44**
by Robert Bloom

applications

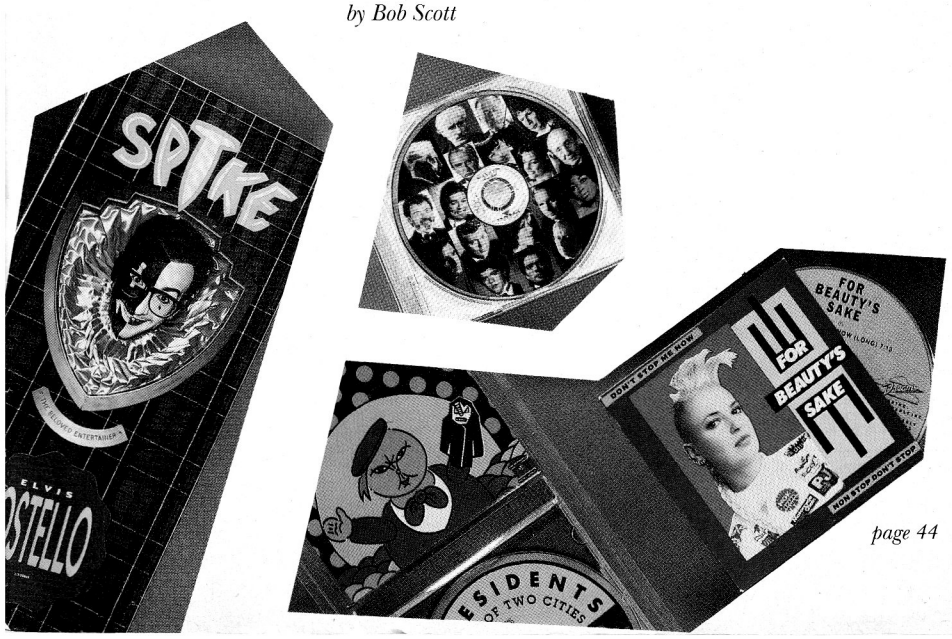
Turbosynth Tips
Are you ready for post-graduate work in Turbosynth? The program's author teaches advanced techniques **23**
by Mark Jeffery

Using Dr. T's Programmable Variations Generator
From the mundane to the miraculous, this program does a lot more than algorithmic composition **28**
by Jim Johnson

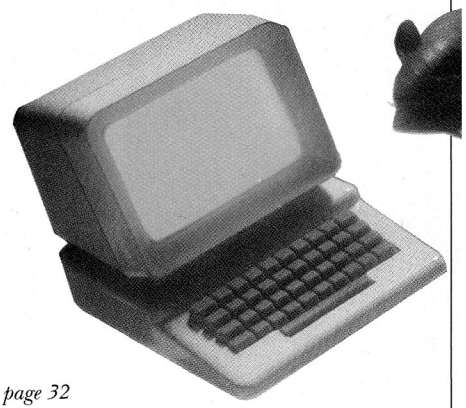
do-it-yourself

RandoM1: A Patch Generator/Librarian for the Korg M1
More than a mere patch librarian, this Atari ST program includes some clever algorithmic functions **48**
by David Snow

Live Film Sync Without SMPTE
If your band uses a sequencer and you want synchronized visuals—without breaking the bank—this is the article for you. **64**
by Bob Scott



page 44



page 32

reviews

First Take: Capsule Comments
Dr. T's KEYS!, Yamaha ARM1 Sliding Rack-mount Adapter **70**

Opcode Vision for the Macintosh *by Craig Anderton* **76**

Sansui WS-X1 6-track Cassette Deck/Mixer/Mixdown Deck
by Steve Oppenheimer **84**

RealTime 1.1 for the Atari ST
by Jamie Krutz **92**

departments

FYI: For Your Information **7**
Letters **8**
What's New. **12**
Service Clinic **58**
Ad Index **90**
Music Re:Views **98**
Classifieds **101**
The Back Page **106**

Cover: Photograph by David Bishop.

Electronic Musician® is published at 6400 Hollis St. #12, Emeryville, CA 94608, and is ©1989 by NBB Acquisitions, Inc. This is Volume 5, Number 8, August 1989. **Electronic Musician** (ISSN: 0884-4720) is published monthly. Second Class postage paid at Oakland, CA, and additional mailing offices. All rights reserved. This publication may not be reproduced or quoted in whole or in part by any means, printed or electronic, without the written permission of the publishers.

RandoM1: A PATCH GENERATOR/ LIBRARIAN FOR THE KORG M1

This useful accessory program for the Korg M1 and Atari ST not only saves and loads patches to and from disk, but generates new sounds to delight your ears.

By David Snow

Original Data Format (seven bytes)								
Byte 0	b7	b6	b5	b4	b3	b2	b1	b0
Byte 1	b7	b6	b5	b4	b3	b2	b1	b0
...Byte 6	b7	b6	b5	b4	b3	b2	b1	b0

MIDI Data Format (eight bytes)								
Byte 0 (msb_byte)	0	b7	b7	b7	b7	b7	b7	b7
Byte 1	0	b6	b5	b4	b3	b2	b1	b0
...Byte 7	0	b6	b5	b4	b3	b2	b1	b0

FIG.1: M1 Data Dump Format.

It was a tough choice: either go on a Hawaiian dream vacation or buy a big black box of metal and plastic (kind of makes you wonder about some people's values, doesn't it?). My wife smashed through the logjam of indecision when she advised me to go ahead and buy the Korg M1; anything was preferable to my constant whining and sulking. Well, I haven't regretted the decision. What an axe! It has 142 waveforms in ROM (4 megabytes worth), including multisamples, drums, and digitally generated sounds, all in sparkling 16-bit fidelity, not to mention two independent, programmable stereo effects units, 8-program multitimbral output with up to 16-voice polyphony, and an 8-track sequencer. It's a nice package. (*For more information on the M1, see the review in the November 1988 issue—Ed.*)

However, programming a synth as sophisticated as the M1 can be intimidating. A random patch generator can help overcome "programmer's inertia" and provide a few jolts of inspiration for jaded ears, but where do you put all those wonderful new programs? There's no internal disk drive for saving program and sequence data, so to make room for your new patches, you'll have to either erase some of the factory patches (most of which are too good to waste), buy a handful of those costly little 256K RAM cards, or save and load data via MIDI system exclusive (sys ex) dumps. Clearly, the latter is the most cost-effective way to go if you already have a computer.

RandoM1 is a GEM-based, integrated patch generator/librarian for the M1 and Atari ST. It stores programs to disk in banks of 50 or 100, depending on which M1 memory allocation option you select. New patches are created by selecting parameters randomly from other patches, a trick Tim Dowty used as the basis for his CZ-101/C-64 patch generator, CZPLUS (August 1987 **EM**).

HOW IT WORKS

To save a bank of patches to disk, or generate a new patch, it is first necessary to transfer patch data from the M1 to the computer. To do this, RandoM1 sends a MIDI system exclusive (sys ex) All Program Parameter Dump Request to the synth, which responds by sending a sys ex dump of all its programs. (For information about system exclusive messages, see Tim Dowty's "CZ Patch Librarian" in the February 1987 **EM** and Jim Johnson's "Fun With System Exclusives" in the March 1987 issue.)

To create a new sound, the software selects 143 parameter bytes randomly from the patches sitting in the ST and stuffs the mongrelized data into a buffer. The new patch then goes to the M1 in a Single Program Parameter Dump, and RandoM1 patiently waits for you to try out the sound on the instrument. If you like what you hear, you can store it in the M1 in the normal fashion (push the synth's EDIT PROG button, go to Page 9, press WRITE, and select OK), or you can keep creating new patches until you find one you like.

Loading the patch bank in the ST and saving it to disk requires binary saves and loads of the bank buffer, while loading the synth with a patch bank requires an All Program Parameter Dump from the ST.

There is a complication, however. According to the MIDI spec, data bytes for any MIDI message (including system exclusive messages) cannot exceed a value of 127. However, many of the M1's parameters exceed 127, so the M1 needs to "compress" its data before sending over MIDI.

The CZ-101 method of compression, accommodated by CZPLUS, splits each byte into two nibbles (4-bit units) and sends them in separate bytes. The M1 uses a more efficient but more complex scheme (**Fig. 1**). It groups the data into sets of seven bytes each, strips off bit 7 of each byte in the set and places it into an extra byte, then sends this byte before the other seven during the dump. Therefore, for every seven bytes of original data, eight are sent via MIDI (and it's a good thing, too, since a patch dump in the CZ sys ex format would take almost

RandoM1: A Patch Generator/Librarian for Korg M1

```
1   ' RANDOM 1 patch generator/librarian for the
2   ' Korg M1 Music Workstation and the Atari ST
3   ' (C) Copyright 1988 by David Snow
4
5   ' LDW BASIC Compiler
6   ' (C) Copyright 1987 by Logical Design Works, Inc.
7
8   defwrd a-z: randomize (0): old%=switch(0): mouse 256
9
10  dim static allprg_bufr(8172): allprg_bufr%=varptr(allprg_bufr(0))
11  dim static dump_module(52): dump_module%=varptr(dump_module(0))
12  dim static send_module(65): send_module%=varptr(send_module(0))
13  dim static progrm_bufr(72): progrm_bufr%=varptr(progrm_bufr(0))
14
15  for index=0 to 102
16       read byte$: poke_b dump_module%+index,val("&h"+byte$)
17  next
18       data 60,1A,00,00,00,4A,00,00,00,00,00,00,00,00,00
19       data 00,00,00,00,00,00,00,00,00,00,FF,FF,20,6F,00,06
20       data 2C,68,00,04,61,00,00,2E,3C,00,3A,3C,00,07,61,00
21       data 00,24,0C,00,00,7F,62,00,00,1A,38,06,EB,6C,02,44
22       data 00,80,88,40,1C,C4,53,45,4A,45,66,00,FF,E2,60,00
23       data FF,D4,4E,75,3F,3C,00,03,3F,3C,00,02,4E,4D,58,8F
24       data 06,40,01,00,4E,75,00
25
26  for index=0 to 128
27       read byte$: poke_b send_module%+index,val("&h"+byte$)
28  next
29       data 60,1A,00,00,00,64,00,00,00,00,00,00,00,00,00
30       data 00,00,00,00,00,00,00,00,00,00,FF,FF,2A,6F,00,06
31       data 2C,6D,00,04,2E,2D,00,0C,61,00,00,0C,2E,2D,00,14
32       data 61,00,00,04,4E,75,42,46,42,45,38,3C,00,07,16,36
33       data 50,00,02,43,00,80,E8,6B,8C,43,52,45,53,44,4A,44
34       data 66,00,FF,EC,61,00,00,1A,38,3C,00,06,1C,1E,02,46
35       data 00,7F,61,00,00,0C,51,CC,FF,F4,51,CF,FF,CA,4E,75
36       data 3F,006,3F,3C,0,03,3F,3C,00,03,4E,4D,5C,8F,4E,75,00
37
38  for index=0 to 3
39       read hex_val$: sysex_header(index)=val("&h"+hex_val$)
40  next
41       data F0,42,30,19
42  for index=0 to 2
43       read hex_val$: dump_request(index)=val("&h"+hex_val$)
44  next
45       data 1C,00,F7
46  for index=0 to 3
47       read hex_val$: prog_mode(index)=val("&h"+hex_val$)
48  next
49       data 4E,02,10,F7
50  for index=0 to 3
51       read hex_val$: edit_mode(index)=val("&h"+hex_val$)
52  next
53       data 4E,03,10,F7
54  for index=0 to 5
55       read char$: prog_name(index)=asc(char$)
56  next
57       data R,A,N,D,O,M
58  patch_number=0: data_sets=2041: data_rem=5: file_len%=14300
59  bank=0: alloc=100: lines=19: dir$="A:\*.M1": sel_file$=""
60
61  alert 3,"Set M1 to MIDI global channel 1",1," OK ",exit_val
62
63  menu 1,1,1,"  Random 1          "
```

LISTING 1: RandoM1 Source Code in LDW Basic.

continued on page 52

twice as long).

RandoM1 reassembles sys ex data into its original form with a machine-code subroutine (**Listing 3**). This routine checks the MIDI port for incoming data and keeps processing that data until it encounters a byte with a value greater than 127, which indicates the end of the sys ex message. The routine is fast enough to process an all-program dump without having to enlarge the ST's 128-byte default MIDI buffer.

To send data back to the M1, either as a single-program or all-program dump, requires converting back into the M1 sys ex format. The subroutine in **Listing 2** accomplishes this and requires three parameters:

- A pointer to the patch data buffer;
- A counter indicating the number of 7-byte data sets to be outputted; and
- A remainder value indicating the number of bytes left over after the last 7-byte set.

Both machine-code modules are loaded from data statements in the RandoM1 program listing (**Listing 1**, lines 15-24 and 26-36).

Have a large music collection?
You need
Personal Music Librarian.

Use your computer and Personal Music Librarian to organize your recorded & sheet music collection.

This easy-to-use software package features user-defined fields, categories, and subcategories. Reporting facilty lets you design labels and reports. Stores albums and selections. Help is available at the press of a key. Retrieval is fast and flexible. Store up to 20 pages of text per entry. The comprehensive manual is supplemented by telephone assistance if you need it.

$55. For IBM PC/compatibles and Atari ST. Coming for Mac and Amiga. Look for demo version on GEnie and CIS. Hours 4-10 PM Eastern M-Th. Phone **404-242-0887**. Visa/MC accepted. Ask for free info packet.

Professional Database Applications for Personal Computers

Personal Database Applications

2634 Meadow Bend Court, Duluth, GA 30136

ACE MUSIC CENTER

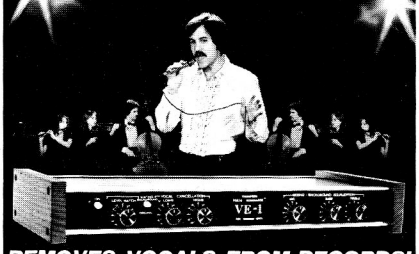
ACE MUSIC CENTER

FOR A FULL LINE OF PROFESSIONAL EQUIPMENT, AT A PRICE YOU CAN AFFORD— CALL **ACE**

1-305-892-6100

13630 WEST DIXIE HWY.
N. MIAMI, FLA. 33161

A SINGER'S DREAM!



REMOVES VOCALS FROM RECORDS!

Our VOCAL ELIMINATOR can remove most or virtually all of a lead vocal from a standard stereo record and leave most of the background untouched! Record with your voice or perform live with the backgrounds. Used in Professional Performance yet connects easily to a home component stereo system. Not an equalizer! We can prove it works over the phone. Write or call for a free brochure and demo record.

Listen..

Before You Buy!

- **Time Delay**
- **Reverberation**
- **Crossovers**
- **Noise Reduction**
- **Compressor/Limiters**
- **Expanders**
- **Spectrum Analyzers**
- **Parametric EQ**

Don't have regrets about paying too much for a lesser product. In demos and comparisons, we'll show you why we're Better! Our Factory Direct sales allow us to produce a Superior product and offer it to you at a Lower price. Call or write for a free full length Demo Album and 24 page brochure.

LT Sound, Dept. EM-9, 7980 LT Parkway
Lithonia, GA 30058 (404) 482-4724
24 HOUR PHONE DEMO LINE: (404)482-2485

LISTING, from page 51

```
  64   menu 2,0,1," File "
  65   menu 2,1,1," Open...   "
  66   menu 2,2,1," Save...   "
  67   menu 2,3,1," Quit      "
  68   menu 3,0,1," Dump "
  69   menu 3,1,1," Synth to computer "
  70   menu 3,2,1," Computer to synth "
  71   menu 4,0,1," Randomize "
  72   menu 4,1,1," Generate patch "
  73   menu 5,0,1," Allocate "
  74   menu 5,1,2,"  100 programs "
  75   menu 5,2,1,"  50 programs "
  76   openw 0,0,0,640,400,0
  77
  78   on dialog gosub dialog_trap: on menu gosub menu_trap
  79   dialog on: menu on
  80   reset: mouse 257
  81
  82   while 1: wend
  83
  84   dialog_trap:
  85        on dialog(0)gosub dummy,dummy,dummy,redraw_window
  86        return
  87
  88   menu_trap:
  89        on menu(0) gosub desk,files,dump,generate,allocate
  90        return
  91
  92        desk:
  93             if menu(1)=1 then alert 1,"RANDOM 1|(C) Copyright 1988 David Snow | |LDW BASIC
(C) 1987|by Logical Design Works, Inc.",1," OK ",exit_val
  94             return
  95        files:
  96             on menu(1) goto open_file,save_file,exit
  97                  open_file:
  98                       gotoxy 0,0: print " Select file to load:          ";
  99                       ask file dir$,sel_file$,exit_file$,exit_val
 100                       if exit_val=0 then return
 101                       bload exit_file$,allprg_bufr%
 102                       mouse 256: gosub print_names
 103                       return
 104                  save_file:
 105                       gotoxy 0,0: print " Enter file to save:           ";
 106                       ask file dir$,sel_file$,exit_file$,exit_val
 107                       if exit_val=0 then return
 108                       bsave exit_file$,allprg_bufr%,file_len%
 109                       return
 110                  exit:
 111                       end
 112        dump:
 113             on menu(1) gosub dump_to_computer,dump_to_synth
 114             return
 115        generate:
 116             gosub random_patch
 117             while 1
 118                  alert 0,"1. Generate another patch  |2. Resend patch|3. Abort| ",1," 1 | 2 |
3 ",exit_val
 119                  if exit_val=3 then return
 120                  on exit_val gosub random_patch,send_patch
 121             wend
 122        allocate:
 123             if menu(1)=1 then
 124                  data_sets=2041: data_rem=5: file_len%=14300
 125                  bank=0: lines=19: alloc=100
 126                  menu 5,1,2: menu 5,2,1
 127             else
 128                  data_sets=1020: data_rem=2: file_len%=7150
 129                  bank=2: lines=9: alloc=50
 130                  menu 5,1,1: menu 5,2,2
 131             endif
 132             return
 133
 134   dump_to_computer:
 135        mouse 256: clearw 0
 136        print " Getting programs from M1..."
 137        while inp(-3)<>0: discard=inp(3): wend
```

```
138       gosub send_header
139       for index=0 to 2: out 3,dump_request(index): next
140       while inp(3)+256<>&hF0: wend
141       for byte=0 to 4: discard=inp(3): next
142       call dump_module%(allprg_bufr%)
143  print_names:
144       clearw 0
145       n=0: pointer%=allprg_bufr%
146       for x=0 to 60 step 15
147            for y=0 to lines
148                 gotoxy x,y: print str$(n);". ";
149                 for index=0 to 9
150                      print chr$(peek_b(pointer%+index));
151                 next
152                 n=n+1: pointer%=pointer%+143
153            next
154       next
155       reset: mouse 257
156       return
157
158  dump_to_synth:
159       mouse 256: reset
160       gotoxy 0,0: print " Sending programs to M1...          ";
161       gosub send_header: out 3,&h4C: out 3,bank
162       call send_module%(allprg_bufr%,data_sets,data_rem)
163       out 3,&hF7
164       redraw 0: mouse 257
165       return
166
167  random_patch:
168       patch_number=patch_number+1
169       for index=1 to len(str$(patch_number))
170            prog_name(index+5)=asc(mid$(str$(patch_number),index,1))
171       next
172       for index=0 to 9
173            poke_b progrm_bufr%+index,prog_name(index)
174       next
175       for param%=10 to 142
176            poke_b progrm_bufr%+param%,peek_b(allprg_bufr%+(int(rnd*alloc)*143)+param%)
177       next
178
179  send_patch:
180       gosub send_header: out 3,&h40
181       call send_module%(progrm_bufr%,19,2)
182       out 3,&hF7
183       gosub send_header
184       for index=0 to 3: out 3,edit_mode(index): next
185       gosub send_header
186       for index=0 to 3: out 3,prog_mode(index): next
187       return
188
189  send_header:
190       for index=0 to 3: out 3,sysex_header(index): next
191       return
192
193  redraw_window:
194       redraw dialog(4)
195       return
196
197  dummy:
198       return
```

USING THE PROGRAM

RandoM1 is a GEM application, which allows access to desk accessories and generally makes the user interface more pleasant. The program is written in LDW BASIC (from Logical Design Works, 780 Montague Expressway #205, San Jose, CA 95131), a compiled language that creates stand-alone program files requiring no run-time module. If you have LDW BASIC, type in the listing with the text editor of your choice (don't bother with line numbers, they're just for reference), save it, compile it (with the "window environment" option off), and run it in high or medium resolution. (*Unlike other ST programs that have appeared in* **EM** *recently,*

EUROPADISK, LTD.

DMM LP MASTERING

State-of-the-art, Direct Metal Mastering, a quantum leap in LP quality

LP PRESSING

DMM audiophile quality—Teldec vinyl

DIGITAL SERVICES

- Neve Digital Console—EQ, Limiting/Compression
- Sony Editing
- Digital Format Conversion
- Analog-to-Digital Transfer
- CD Master Tapes
- DAT Copies

COMPACT DISC PRODUCTION
CASSETTE DUPLICATION
7 INCH SINGLES

Complete packages with printing

EUROPADISK, LTD.
75 Varick Street
New York, NY 10013
(212) 226-4401

IEMUG

NO on-line fees for over *50 Megs of MIDI!*

• Programs • Songs
• Patches • Help
• Classifieds
• Fellow MIDI-ists!!

Central Host	405/733-3102	3/12/24
NYC	212/751-2347	3/12/24
San Diego	619/698-7155	3/12/24
Columbus, OH	614/848-5947	3/12/24/96
Cincinatti 1	606/727-3638	3/12/24/96
Cincinatti 2	513/396-7166	3/12/24/96
Greensboro, NC	919/274-5760	3/12/24/96
Phoenix, AZ	602/864-0244	3/12
Irvine, CA	714/552-3036	3/12/24/96
Montreal, Canada	514/744-7354	3/12
Hong Kong	852/369-8647	3/12/24

and over 400 other BBS's.
Call for nearest affiliate!

International Electronic Musician's Users Group

Voice: 405/736-6676

Son of FX™

ORGANIZE YOUR RECORDING, COMPUTER AND RACK-MOUNTABLE EQUIPMENT IN AN ATTRACTIVE, MOBILE WORK STATION.

SON OF FX FEATURES:
- Two adjustable shelves for multi-tracks, drum machines, computers and more!
- Holds up to 10 rack "spaces" of equipment.
- Shown with optional "Rack Drawer™"

ASK YOUR DEALER FOR A DEMONSTRATION
CALL OR WRITE FOR FREE BROCHURE

FOUR DESIGNS COMPANY 6531 GROSS AVENUE • CANOGA PARK, CA 91307 (818) 716-8540 OUTSIDE CA 800-544-3746

FOR ADDITIONAL FOUR DESIGNS PRODUCT INFORMATION, CALL:
1-800-544-3746
SAY YOU SAW IT IN
ELECTRONIC MUSICIAN!

FX RACK

ECONOMICAL 19" RACK MOUNT "FURNITURE" FOR THE PROFESSIONAL OR HOME STUDIO.

FX RACK FEATURES:
- Holds up to 18 "spaces" of equipment.
- Attractive black textured vinyl laminate.
- Shown with optional wheels.

ASK YOUR DEALER FOR A DEMONSTRATION
CALL OR WRITE FOR FREE BROCHURE

FOUR DESIGNS COMPANY 6531 GROSS AVENUE • CANOGA PARK, CA 91307 (818) 716-8540

grandma's music & sound

Large Inventory
Major Brands
Solid Technical Advice
Factory Service Center
Price Information
505-292-0341
Customer Support and Orders
800-444-5252

800 Juan Tabo NE
Albuquerque, NM 87123

• M1 PATCH GENERATOR

RandoM1 is too complex to convert to ST BASIC, so you must have LDW BASIC in order to use this program—Ed.)

A dialog box will ask you to set the M1's global MIDI channel to channel 1. Also, make sure that the M1 can send and receive system exclusive information (as determined by the EXCL switch on Global Page 5-2). RandoM1's default setting is for large program allocation (100 programs/100 combinations). To set up for large sequence allocation (50 programs/50 combinations), point to the Allocate heading on the menu bar and click on "50 programs."

To load the ST with the M1's patch bank, point to the Dump menu heading and click on Synth to Computer. The program then names and numbers all programs in the bank, on screen. Be sure to save the bank to disk *before* you begin messing around with your patches. Enter a file name in the file selector box (the default file extension is .M1), then press RETURN.

To create a new patch, point to the Randomize menu and click on Generate Patch. To help you keep track of these patches, each patch is named RANDOM when created, followed by a number from 1 to 999. Try the patch out on the M1 keyboard and, if you like it, save it in the synth. If you don't like it, press RETURN, or click on button 1 in the dialog box to generate another sound.

On rare occasions the transfer of a new patch from the ST to the M1 will glitch (evidenced by the RANDOM patch number on the synth's LCD not incrementing). If this happens, resend the patch by clicking button 2 of the dialog box. Resending is also useful if you start to edit a generated patch at the M1, mess it up, and want to start over. When you've had your fill of patch generation, click on button 3.

Load a patch bank disk file into the ST. You can dump that patch bank to the M1 by pointing to the Dump menu and clicking on Computer to Synth, or you can use the bank to generate new patches.

M1 Sysex Send Module Assembly

```
            move.l 6(a7),a5          get parameter pointer from stack
            move.l 4(a5),a6          get data buffer pointer a6
            move.l 12(a5),d7         get loop counter (number of data sets)
            bsr loop1                convert to sysex format
            move.l 20(a5),d7         get loop counter (bytes left over)
            bsr loop1                convert to sysex format
            rts                      return to calling program
loop1       clr.w d6                 clear msb_byte
            clr.w d5                 reset address index
            move.w #7,d4             set bit-shift counter
loop2       move.b 0(a6,d5.w),d3     get byte from buffer
            andi.w #128,d3           isolate bit 7
            lsr.w d4,d3              shift right
            or.w d3,d6               update msb_byte
            addq.w #1,d5             increment address index
            subq.w #1,d4             decrement bit-shift
            tst d4                   last byte in set?
            bne loop                 if not, process next byte
            bsr bconout              else output msb_byte to MIDI
            move.w #6,d4             set byte counter (7 bytes/data set)
loop3       move.b (a6)+,d6          get byte and increment pointer
            andi.w #127,d6           clear bit 7
            bsr bconout              output to MIDI
            dbra d4,loop3            last byte in set?
            dbra d7,loop1            if so, process next data set
            rts

bconout     move.w d6,-(a7)
            move.w #3,-(a7)
            move.w #3,-(a7)
            trap #13
            addq.l #6,a7
            rts

            end
```

LISTING 2: Sysex Send Module.

SMART
UPGRADE

Break the barriers! Announcing the CMS-404 interface.

For unlimited MIDI power on your PC. The ideal low-cost add-on to your 401-based system.

When used with a 401 interface, you get two MIDI ports in and five out–a total of 80 channels! Works with other programs supporting CMS multiport interfaces, such as 64 Track PC, Cakewalk, Texture, and more to come.

FREE (UNTIL MAY 1): PC Sequencer software included, featuring 600 pulses/quarter note and piano roll graphics. Also compatible with the CMS-404 and all 401 interfaces.

Only $199 from Computer Music Supply, 382 N. Lemon Ave., Walnut, CA 91789. We accept VISA, MasterCard and AMEX.

© 1989 Music Creator Just For You, Inc. Patent Pending
Call 800-322-MIDI.
Computer Music Supply
382 N. Lemon · Walnut, CA 91789 · 714-594-5051

CMS

M1 Sysex Receive Module Assembly

```
                move.l 6(a7),a0          get parameter pointer from stack
                move.l 4(a0),a6          get data buffer pointer a6
loop1           bsr bconin               get msb_byte from MIDI port
*                                        (msb_byte is the bit-7
*                                        holder for each data set)
                move.w d0,d6             save it
                move.w #7,d5             set loop2 counter (7 bytes/set)
loop2           bsr bconin               get next byte from MIDI port
                cmpi.b #127,d0           end of sysex?
                bhi return               if so, return to calling program
                move.w d6,d4             retrieve msb_byte
                lsl.w d5,d4              shift bit 7
                andi.w #128,d4           isolate it
                or.w d0,d4               add lower bits (byte complete)
                move.b d4,(a6)+          store byte and increment pointer
                subq.w #1,d5             test for last byte of data set
                tst d5
                bne loop2                if not, get next byte of set
                bra loop1                else start next set
return          rts

bconin          move.w #3,-(a7)          get data from MIDI port
                move.w #2,-(a7)
                trap #13
                addq.l #4,a7
                addi.w #256,d0           convert to unsigned value
                rts

                end
```

LISTING 3: Sysex Receive Module.

HELPFUL HINTS

After generating a patch, play the M1's keyboard in all registers before you decide to keep or trash it; some patches sound lousy in one range and great in another. Remember that a potentially interesting patch might have a long attack slope, so hold the key down if you don't hear anything at first.

If you're accustomed to the factory programs, you'll be intrigued and, occasionally, amused by the generated patches. Some will sound quite natural and others quite unnatural. Many will be usable with some tweaking, depending on your taste and sanity.

Although all random patch parameters are derived from other patches, it is still possible to have "illegal" values since they are taken out of context. Sometimes a patch will not play at all (usually when the oscillator mode is set to DRUMS), or more rarely, will produce distinctly unmusical weirdness, which seems to happen when an Early Reflections program has been selected in the M1's effects section; the E/R time value is usually bogus. At any rate, keep the output level low enough to avoid alienating neighbors and loved ones.

I like this program because it's full of surprises. It doesn't do everything I'd like, but tradeoffs were necessary (such as being limited to MIDI channel 1) to keep the listing shorter than the Manhattan phone book. You can't shuffle individual patches between bank files and the synth, you can accidentally wipe out your M1 patch bank by dumping an empty buffer to it from the ST (I told you to back up your patches, didn't I?), the program does not check for dump load errors from the M1, there's no error-checking for file operations (trying to open a nonexistent file or attempting to save to a full or write-protected disk will crash the program), and the program hangs if you attempt a dump from the M1 without the synth attached.

But you need something to work on during those inevitable long winter nights, don't you?

David Snow *is a composer and self-styled domestic god (read "househusband"). His music has been premiered by such diverse ensembles as the Composer's Chamber Orchestra, the Harvard Wind Ensemble, the Ruby Shang Dance Company, and the Los Angeles Tuba Quartet.*