Disk Atari 32k

MusicBox

By Jerry White

P.O. Box 131
Marlboro, NJ 07746

(201) 431-3472

Programmers: Jerry White
Craig Patchett
Charles Bachand

Musical Arrangements: Jerry White
Jenny Tesar

Documentation: Jerry White

Printed in the U.S.A.

Contents

Introduction

MusicBox enables a programmer to take music and enter it into his or her own computer program. The music thus entered will run independently of BASIC. For example, *MusicBox* enables you to have background music during a game—without it having any effect on the game.

The disk consists of a menu and seven programs written in ATARI BASIC and machine language. To use *MusicBox*, you need the ATARI BASIC cartridge, a minimun of 32K RAM, and one ATARI 810 disk drive. ATARI's *Music Composer* cartridge is needed if you wish to create your own *MusicBox* songs. A joystick and printer may also be used but are not required.

The documentation assumes that you have some knowledge of ATARI BASIC and the ATARI *Music Composer*. No knowledge of Assembler Language is required. We recommend that you read the following reference manuals:
ATARI BASIC
Disk Operating System II
ATARI Music Composer

How To Use MusicBox

Your *MusicBox* Master Disk contains ATARI's DOS 2.0S. It is important that this version of DOS be used when running any of the *MusicBox* programs. When in doubt, insert this disk into Disk Drive 1, then turn on your computer.

When reading or writing data files, Disk Drive 1 will always be assumed by all the programs in this package.

After a short introductory program that displays the Program Design logo, the MENU program will run automatically. Before using any other program, select menu option 7 to load DUP.SYS. Then remove the *MusicBox* Master Disk.

It is important that you use the Master Disk only as a backup copy. Create a Working Disk by formatting a blank disk via DOS Option "I". Then use DOS Option "J" to duplicate the Master Disk (SOURCE DISK) onto your Working Disk (DESTINATION DISK). Refer to your *Disk Operating System II Reference Manual* for further information.

After creating a Working Disk, store the Master Disk in a safe place. With your Working Disk in Disk Drive 1, power off then power on the computer. If all went well, the introduction and MENU programs will run automatically.

Program Listings: The programs on the disk are not protected, so that they may be studied by the user. The BREAK key has not been disabled (except in the Colorgan program) and hitting the SYSTEM RESET key will not cause the computer to reboot.

Program listings have been provided for your convenience. Listings of *Music Composer* files may be printed using the Translat program. If listings of *MusicBox* files are desired, the DOS Copy option is recommended. To copy a *MusicBox* data file to your printer, select DOS Option "C," then respond to the FROM-TO prompt as follows:

FILENAME.MBD,P:

Exiting from a Program: Some of the programs alter key memory pointers. It is therefore important to exit from a program by using the program's RUN"D:MENU" option. If you wish to abort and list a program, use the SYSTEM RESET key, which will reset most memory pointers.

Altering a Program: If you decide to alter one of the programs, **stop the music** before saving your version. Also: be sure to use a different program name.

Song Data Files: It is important to understand the difference between song data files with the extensions .MUS and .MSB. .MUS indicates a file created by ATARI's *Music Composer*. .MSB indicates a file created by the Convert program in this package.

.MUS files are used as input to the Translat and Convert programs only. .MSB files are used as input to the Playtest and Colorgan programs only.

The MusicBox Programs

Menu: This program runs automatically when you boot the computer using the *MusicBox* disk. Its function is to make program selection as simple as pressing a single key. All other programs on the disk contain an option to return to MENU.

Convert: This program converts data files generated by ATARI's *Music Composer* into a format recognized by the Colorgan and Playtest programs.

Colorgan: This program reads disk files created by the Convert program. A colorful display ("color organ") is generated and altered as each note of your song is played.

Playtest: This program also reads files created by the Convert program. Since the music is played during vertical blank interrupts, you can actually add your own lines of BASIC as the music plays on.

Musicbox: This program displays each note within *Music Composer's* range using a "player." The player may be moved using the UP ARROW or DOWN ARROW keys, or by using a joystick. In each position on the music staff, the actual note is played and the screen displays the note's *Music Composer* representation and its BASIC pitch value. The program uses treble and bass clefs and notes, in the form of a character set, that may be adapted to programs of your own.

Translat: This program was donated by *A.N.A.L.O.G. 400/800* Magazine and modified by Jerry White. It reads files generated by *Music Composer* and optionally prints or displays these files in DATA entry format.

Data Files: The *MusicBox* disk includes several *Music Composer* and Musicbox song data files. Those with the filename extention .MUS were created using *Music Composer*. Those with the extention .MBD were CONVERTed into Musicbox Data format. When creating your own files, these filename extentions should be used.

Convert Program

Before you can have your own *Music Composer* files played by the Playtest or Colorgan programs, they must be converted from *Music Composer* format (.MUS) to *MusicBox* format (.MBD). By the way, "MBD" stands for MusicBox Data.

The Convert program will display conditions it cannot handle as the conversion is made. For example, later on you will be specifying a decay rate for the volume of each voice. Therefore, Convert will ignore *Music Composer* volume commands and display them as BYPASSED ERRORS.

Convert will not permit nested GOTO loops and will not compensate for TRANSPOSE errors that are out of *Music Composer's* range.

This program assumes that all *Music Composer* files contain four voices. If a voice is not used, use *Music Composer* to enter a VOLUME 0 command for the voice. This is done using the ARRANGE feature. See your *Music Composer* documentation for further details.

CONVERT is written in pure ATARI BASIC and must make extensive calculations. It may take a few minutes to convert each file. We have included an option to speed up this process by about one third. The initial display will explain this and will ask that you type:

1—for constant error update
2—to blank the screen during the conversion and turn it on again at the finish

The second option will increase conversion speed but you may lose lines of errors that could scroll off the screen. If you think you've lost error lines and need to see them, you can reconvert the file later on, using the constant error update display.

When you create a *Music Composer* file, SAVE it on your *MusicBox* Working Disk. You may also use DOS to duplicate your *Music Composer* files onto the Working Disk.

The *Music Composer* file is assumed to be on Disk Drive 1. The converted *MusicBox* data file will be written on Disk Drive 1. The program will ask for the *Music Composer* filename.

CONVERT will then read that file into memory. You will be asked to specify the NEW MUSICBOX FILENAME (which will automatically be given a .MBD extention by the program). Be careful: if you specify a *MusicBox* filename that matches one already on the disk, the old file will be lost. Keep track of all data files! A directory of all files on a disk may be displayed using MENU Option #1 or DOS Option A.

After you have given the file a name, you will have the opportunity to specify the starting and ending volume for each voice. Note that volumes must be specified even if a voice is not used. This provides a crude level of sound shaping. The only way to find out how certain volumes affect a song is by playing the song—that is, by trial and error.

You will also be able to specify whether or not you would like breaks between notes. Let's assume that VOICE 1 contains the main melody and you want it to play a bit louder than the other voices. Try converting your file specifying Starting Volume (SV) of 8 and Ending Volume (EV) of 6 for Voice 1, and SV = 4, EV = 2 for all other voices. Choose Option 1—no breaks between notes—for all voices. The result may not be quite what you want, so that you will have to try different volumes and perhaps add breaks. As we said, trial and error will be the best teacher.

When a conversion is completed, your new *MusicBox* file will be written onto Disk Drive 1 and the MENU program will be reloaded automatically. It is recommended that you now test the sound of your new file using the Playtest program.

```
0 POKE 559,0:GOSUB 30000:REM CONVERT [5/22/82]
1070 D$="":FOR X=00 TO 037:READ A:PITCH(X)=A:NEXT X:FOR X=00
 TO 64 STEP 04:FOR Y=00 TO 02:READ A:CROSS(X+Y)=A:NEXT Y:CRO
SS(X+06)=00:NEXT X
1120 ? "R":GOSUB 0720:AD=ADR(FI$)-01
1130 GET #01,A:IF A=0255 THEN GOTO 0270
1140 IF A<>170 THEN GOTO 0130
1150 GET #01,A:IF A<0128 THEN 1170
1160 GET #01,A:GET #01,A:GET #01,TEMPO:GET #01,A:GET #01,A:G
OTO 0130
1170 IF A<020 THEN 1210
1180 VOICE=(A-18)/02:PO=050*VOICE+01
1190 GET #01,A:IF A=0255 THEN V$(PO,PO)=CHR$(A):GOTO 0130
1200 GET #01,B:V$(PO,PO)=CHR$(A):V$(PO+01,PO+01)=CHR$(B):PO=
PO+02:GOTO 0190
1210 PHRASE=A/02:PO=LP*PHRASE+01
1220 GET #01,A:IF A=0255 THEN P$(PO,PO)=CHR$(A):GOTO 0130
1230 GET #01,B:B=B-01:IF A=0127 THEN GOTO 0220
1250 IF A>85 THEN GOTO 0190
1260 P$(PO,PO)=CHR$(CROSS(A)):P$(PO+01,PO+01)=CHR$(B):PO=PO+
02:GOTO 0220
1270 TRAP 0270:FN$="D1:":D$="":? :? :? :? "ENTER NEW MUSIC
BOX FILENAME":? :? ">";:CLOSE #01:P=01:TRAP 40000
1271 GOSUB 32000:INPUT D$:LD=LEN(D$):IF LD=00 THEN GOSUB 322
00:GOTO 0270
1272 TRAP 10000:FOR ME=01 TO LD:IF D$(ME,ME)="." THEN POP :G
OTO 1278
1273 NEXT ME
1274 IF LD>08 THEN ME=9:GOTO 1278
1276 D$(LEN(D$)+1)=".MBD":LD=LD+04:GOTO 1279
1278 D$(ME,ME+3)=".MBD":LD=LEN(D$)
1279 TRAP 12000:FN$(LEN(FN$)+01)=D$:OPEN #01,08,00,FN$:TRAP
07
1280 ? "R↓":FOR V=01 TO 04
1281 ? "VOICE ";V,"STARTING VOLUME";:TRAP 1281:INPUT A:A=INT
(A):S(V)=A:IF A<01 OR A>15 THEN 1281
1282 ? ,"FINAL VOLUME";:TRAP 1282:INPUT B:B=INT(B):IF B>15 O
R B<00 THEN 1282
1283 F(V)=B:TRAP 1283:? "ENTER 1 FOR NO BREAK BETWEEN NOTES
OR":? "ENTER 2 FOR BREAKS";:INPUT C
1284 C=INT(C):IF C<01 OR C>02 THEN 1283
1285 B(V)=C:? :NEXT V:? "R CONVERSION ERRORS BYPASSED:":TRAP
 31000
1290 IF CH=02 THEN D1=PEEK(0286):D2=PEEK(0774):D3=PEEK(0272)
:POKE 0286,00:POKE 0774,00:POKE 0272,00
1300 FOR V=01 TO 04:L=00:TR=00:POKE AD+P,010:POKE AD+P+01,F(
V):POKE AD+P+02,S(V):POKE AD+P+06,B(V):P=P+04:PO=050*V+01
1310 A=ASC(V$(PO,PO)):B=ASC(V$(PO+01,PO+01)):IF A=0255 THEN
1330
1320 ON A+01 GOTO 1330,1340,1370,1480,1490,1500,1510
1330 PO=PO+02:GOTO 0310
1340 COUNT=COUNT-01:IF COUNT=00 THEN PO=PO+02:GOTO 0310
1350 IF COUNT<00 THEN ? "RENDLESS LOOP IN VOICE ";V;" IGNORE
```

```
D":PO=PO+Q2:GOTO Q310
1360 PO=Q50*V+Q1+(B-Q1)*Q2:GOTO Q310
1370 POS=LP*B+Q1
1380 IF ASC(P$(POS,POS))=Q255 THEN PO=PO+Q2:GOTO Q310
1390 C=ASC(P$(POS,POS)):D=ASC(P$(POS+Q1,POS+Q1)):POS=POS+Q2:
IF C=Q0 THEN C=C-TR
1410 IF C+TR<Q0 OR C+TR>Q37 THEN ? "■TRANSPOSE ERROR IN VOIC
E ";V:C=-TR+Q1+36*(C+TR>Q37)
1420 POKE AD+P,PITCH(C+TR):P=P+Q1:IF D>Q127 THEN D=D-Q128
1440 BT=INT(D/Q2):D=(D/Q2=BT):TIME=INT((Q2^(BT+Q1))+Q9)*TEMP
O:IF D=Q0 THEN TIME=TIME*1.5
1460 IF TIME>Q255 THEN POKE AD+P,Q255:POKE AD+P+Q1,PITCH(C+T
R):P=P+Q2:TIME=TIME-Q255:GOTO 1460
1470 POKE AD+P,TIME:P=P+Q1:L=L+Q2:GOTO 1380
1480 TR=TR+(B-Q128*(B>Q127))*-(B>Q127):PO=PO+Q2:GOTO Q310
1490 ? "■VOLUME CHANGE IN VOICE ";V;", LINE ";(PO-Q50*V-Q1)/
Q2+Q1:PO=PO+Q2:GOTO Q310
1500 PO=PO+Q2:GOTO Q310
1510 IF COUNT>Q0 THEN ? "■NESTED LOOP IN VOICE ";V;", LINE "
;(PO-Q50*V-Q1)/Q2+Q1
1520 COUNT=B:PO=PO+Q2:GOTO Q310
1530 IF L<Q2 THEN POKE AD+P-Q4,14:POKE AD+P,Q0:POKE AD+P+Q1,
Q255:P=P+Q2
1540 POKE AD+P,Q255:POKE AD+P+Q1,Q255:P=P+Q2:NEXT V:P=Q1:FOR
 V=Q1 TO Q4:S(V)=Q0:P=P+Q4:F(V)=Q0
1570 A=PEEK(AD+P):B=PEEK(AD+P+Q1):P=P+Q2:IF A=Q255 AND B=Q25
5 THEN NEXT V:GOTO 1600
1580 S(V)=S(V)+Q2:F(V)=F(V)+B:GOTO 1570
1600 IF CH=Q2 THEN POKE Q286,D1:POKE Q774,D2:POKE Q272,D3
1605 ? :FOR X=Q1 TO Q4:? "VOICE ";X;" TOTAL TIME ON = ";
1606 IF F(X)=255 THEN ? "0":GOTO 1608
1607 ? F(X)
1608 NEXT X
1610 FOR X=Q1 TO Q4:IF F(X)=F(X+Q1-Q4*(X+Q1=Q12)) OR S(X)<Q4
 OR S(X+Q1-Q4*(X+Q1=Q12))<Q4 THEN NEXT X:P=Q1:MOST=F(Q1):GOT
O 1650
1620 ? :? "■ADJUSTING":MOST=Q1:FOR X=Q2 TO Q4:IF F(X)>F(MOST
) THEN MOST=X
1630 NEXT X:MOST=F(MOST):P=Q1
1650 ? "■WRITING ";FN$:TRAP Q270:CLOSE #Q1
1655 OPEN #Q1,Q8,Q0,FN$:FOR X=Q1 TO Q4:FOR Y=Q1 TO Q4:PUT #Q
1,PEEK(AD+P):P=P+Q1:NEXT Y:TRAP 1710
1660 A=PEEK(AD+P):B=PEEK(AD+P+Q1):P=P+Q2:IF A<>Q255 OR B<>Q2
55 THEN PUT #Q1,A:PUT #Q1,B:GOTO 1660
1670 IF F(X)=MOST OR S(X)<Q4 THEN 1710
1680 DIF=MOST-F(X)
1690 IF DIF>Q255 THEN PUT #Q1,Q0:PUT #Q1,Q255:DIF=DIF-Q255:G
OTO 1690
1700 PUT #Q1,Q0:PUT #Q1,DIF
1710 PUT #Q1,Q255:PUT #Q1,Q255:NEXT X:CLOSE #Q1:GOSUB 32000
1712 ? "■":POKE 752,Q1:? :? ,"CONVERSION COMPLETED":? :? ,"
   LOADING MENU":RUN "D:MENU"
1720 CLOSE #Q1:TRAP Q720:? :? "▶DISK DRIVE 1 IS ASSUMED":FN$
```

```
="D:":D$=""
1721 ? :? "▶ENTER MUSIC COMPOSER FILENAME":? :? "▶";:GOSUB 3
2000:INPUT D$
1722 LD=LEN(D$):IF LD=0 THEN GOSUB 32100:GOTO 1720
1724 TRAP 11000:FOR ME=1 TO LD:IF D$(ME,ME)="." THEN POP :G
OTO 1728
1725 NEXT ME
1726 IF LD>8 THEN ME=9:GOTO 1728
1727 D$(LEN(D$)+1)=".MUS":LD=LD+4:GOTO 1730
1728 D$(ME,ME+3)=".MUS":LD=LEN(D$)
1730 TRAP 1750:FN$(LEN(FN$)+1)=D$:CLOSE #1:OPEN #1,4,0,F
N$:? :? "▶WORKING";:RETURN
1750 ? "■";? :? FN$;" NOT FOUND":GOTO 1720
10000 ? :? "■NEW FILE MUST HAVE .MBD EXTENTION.":GOTO 1270
11000 ? :? "■NEW FILE MUST HAVE .MUS EXTENTION.":GOTO 1721
12000 ? :? "■I WAS UNABLE TO OPEN ";FN$:GOTO 0270
20000 DATA 0,243,230,217,204,193,182,173,162,153,144,136,128
,121,114,108,102,96,91,85,81,76,72,68,64,60,57,53,50,47
20100 DATA 45,42,40,37,35,33,31,29
20200 DATA 1,2,1,3,4,2,5,6,4,6,7,5,8,9,7,10,11,9,12,13,11,13
,14,12,15,16,14,17,18,16,18,19,17,20,21,19,22,23,21
20300 DATA 24,25,23,25,26,24,27,28,26,29,30,28,30,31,29,32,3
3,31,34,35,33,36,37,35,37,0,0
30000 A0=0:A1=1:A2=2:A4=4:A6=3:A7=31000:A8=8:A9=0.5:A10=10:A
12=5:A15=12:A16=6:A20=20:A37=37:A50=50:A72=3072
30010 A130=1130:A190=1190:A220=1220:A255=255:A270=1270:A272=
54272:A286=54286:A310=1310:A720=1720:A774=53774
30020 GRAPHICS A0:POKE 559,A0:POKE 710,A0:DIM FN$(15):FN$="D
1:":A127=127:A128=128
30030 DIM D$(A20),PITCH(40),CROSS(88),V$(250),FI$(A72),S(A4)
,F(A4),B(A4):FI$(A1)="♥":FI$(A72)="♥":FI$(A2)=FI$:V$=FI$
30035 LP=INT((FRE(A0)-2000)/A10):DIM R$(A20),P$(A10*LP):P$(A
1)="♥":P$(LP*A10)="♥":P$(A2)=P$
30040 POKE 82,A1:POKE 83,38:? :? " MUSIC COMPOSER TO MUSIC B
OX DATA FILE"
30045 POKE 201,A10:? ,"■C O N V E R T E R■"

30050 ? :? :? ,"   Please choose:":? "↓▶1. Constant error up
date":? "▶2. Errors displayed at finish"
30060 ? "↓Please note that choice 2 will result in a faster
execution time, but if"
30070 ? "there are more than a screen full of  errors, some
will be lost (since the"
30075 ? "screen will be blank until I'm done)."
30080 ? "I'll ring the bell to let you know":? "each time I
bypass an error."
30085 ? "In either case, please be patient.":? "I'll be very
 busy for a while.":POKE 559,34
30090 TRAP 30090:? :? ,"Your choice";:GOSUB 32000:INPUT CH:T
RAP A7:CH=INT(CH):IF CH<A1 OR CH>A2 THEN 30095
30092 ? "■":RETURN
30095 ? :? ,"■PLEASE TYPE 1 OR 2":GOTO 30090
31000 IF CH=A2 AND PEEK(A286)=A0 THEN POKE A286,D1:POKE A774
```

```
,D2:POKE 0272,D3
31010 POKE 559,34:? "MAKERROR ";PEEK(195);" PLEASE CHECK FIL
E AND  RUN AGAIN":END
32000 FOR ME=010 TO 00 STEP -0.5:SOUND 00,00,02,ME:NEXT ME:R
ETURN
32100 CLOSE #3:OPEN #3,016,00,"D:*.MUS":GOTO 32300
32200 CLOSE #3:OPEN #3,016,00,"D:*.MBD"
32300 ? "R":POKE 752,01:TRAP 32400:? ,"FILES AVAILABLE:":?
32310 INPUT #3,R$:? ,R$:GOTO 32310
32400 ? :? "PRESS ANY KEY TO CONTINUE":POKE 764,0255
32500 IF PEEK(764)=0255 AND PEEK(53279)=7 THEN 32500
32600 POKE 764,0255:POKE 752,00:RETURN
32700 REM
32702 REM ORIGINAL PROGRAM BY CRAIG PATCHETT
32704 REM UPDATED/MODIFIED BY JERRY WHITE
```

Colorgan Program

The Colorgan ("Color Organ") program is an extended version of Playtest. Additional machine language routines have been added to interpret frequency and volume. The routines use this data to change colors on the screen.

This program will ask you to supply the name of the *MusicBox* data file you wish to play and display. As in the Playtest program, you just press the RETURN key at this prompt, to display all *MusicBox* (.MBD) files found on the disk. Remember: Disk Drive 1 is assumed.

Once the file has been read into memory, and the music begins, press any key to exit. You will then have two options. If you press the OPTION key, you will return to the MENU program. If you press the START key, the Colorgan program will rerun itself so that you can enter the name of another *MusicBox* file to play.

The rate of decay specified in the Convert program is what determines color brilliance. The greater the volume, the brighter the colors. As the sound decays, the colors become darker.

Experimentation with the program code is recommended only for advanced programmers. Colorgan contains quite a few REMarks that label the various routines. For example, Colorgan uses Graphics Mode 19 (3 + 16). It will also work using Mode 21 or 23. To try one of these modes, change the GRAPHICS command in line 570. The routine that fills the screen is a subroutine that is found in line 120. Since GRAPHICS 21 and 23 are higher in resolution, the existing fill routine would fill only the upper lefthand corner of the screen. If GRAPHICS 21 is

used, change the "Y" loop TO 47 and the "X" loop TO 79. If GRAPHICS 23 is used, change the "Y" loop TO 95 and the "X" loop TO 159.

The BREAK key has been disabled in this program. Press the SYSTEM RESET key to "break" into this program.

```
0 REM COLORGAN by Jerry White 5/22/82
1 REM
110 GOSUB 310:POKE 559,0:GOTO 150
120 FOR Y=0 TO 23:FOR X=0 TO 39:COLOR X+Y:PLOT X,Y:NEXT X:NE
XT Y:RETURN
130 FOR X=1537 TO 1591:POKE X,0:NEXT X:A=PEEK(106):RETURN

140 REM SETUP TO READ MUSIC INTO RAM
150 RMUSIC=ADR("hh:Uh:T K♥ K♥ K♥ K♥ K♥IJp/ K♥┘ÇS K♥IJPI♦")

160 GBYTE=ADR("█┼)\Ub┘)♥UH┘UI┘ Vdo █♥TTfTP fU♦) )♦")
170 POKE 205,76:L=206:V=GBYTE:GOSUB 680
180 FOR X=1 TO 4:A(X)=E:E=USR(RMUSIC,E):POKE E-2,0:POKE E-1,
0:NEXT X:CLOSE #1
190 REM SETUP SOUND CHANNELS
200 FOR X=0 TO 3:SOUND X,0,10,0:NEXT X
210 REM ENABLE SCREEN & GOTO DLI SETUP
220 GRAPHICS 19:REM USE GRAPHICS MODE 19, 21, OR 23.
222 REM DISABLE BREAK KEY THEN FILL SCREEN WITH COLOR BLOCKS
225 POKE 16,64:POKE 53774,112:GOSUB 120
230 X=USR(ADR(E$),A(1),A(2),A(3),A(4),ADR(A5$)):REM START TH
E MUSIC
240 POKE 764,255:REM WAIT FOR KEY PRESS
250 IF PEEK(764)=255 THEN 250
280 REM STOP THE MUSIC AND EXIT
290 X=USR(ADR(E$)+92):POKE 764,255:POKE 106,RAMTOP
291 COLOR 0:X=0:Y=0:X1=39:Y1=23:PLOT X,Y:FOR I=1 TO 12:DRAWT
O X1,Y:Y=Y+1:DRAWTO X1,Y1:X1=X1-1:DRAWTO X,Y1:Y1=Y1-1
292 DRAWTO X,Y:X=X+1:FOR JW=1 TO 20:NEXT JW:NEXT I:GOTO 800

300 REM SETUP AND INTRO DISPLAY
310 RAMTOP=PEEK(106):DIM A(4),E$(115),PLAY$(15),USER$(15),A5
$(201),DR$(20),DLI$(40):PLAY$="D:"
320 GRAPHICS 0:SETCOLOR 2,9,0:POKE 752,1:POKE 82,1:POKE 83,3
9:? :? ,"   COLOR ORGAN":?
330 ? :? "    Once the music begins to play,":? :? " press a
ny key to return to the MENU."
340 ? :? " The input disk file to this program":? :? " must
be a CONVERTED MUSIC COMPOSER"
350 ? :? "    file with an .MBD extention.":? :? "      Disk
 drive 1 is assumed."
360 ? :? :? " Just press RETURN to list MBD files":POKE 752,
0:POKE 764,255
370 ? :? "    or enter disk filename";:POKE 53279,0:POKE 5327
9,0:INPUT USER$:LU=LEN(USER$):IF LU=0 THEN 730
380 TRAP 710:FOR ME=1 TO LU:IF USER$(ME,ME)="." THEN POP :GO
```

```
TO 388
382 NEXT ME
384 IF LU>8 THEN ME=9:GOTO 388
386 USER$(LEN(USER$)+1)=".MBD":LU=LU+4:GOTO 390
388 USER$(ME,ME+3)=".MBD":LU=LEN(USER$)
390 IF USER$(LU-3,LU)<>".MBD" THEN 710
400 PLAY$(LEN(PLAY$)+1)=USER$:POKE 752,1
410 TRAP 690:CLOSE #1:OPEN #1,4,0,PLAY$:TRAP 40000
420 E$(1,80)="h♥♥h /  / h /  / K K r/ H K - / H K  / H
K / h h  P K ♥ & ( / ' ) ♥ "
430 E$(81,115)="P h h / \d h d a / \d  ♥ Q P "
440 A$(1,90)="♥ P n r/ 8 / i  / i ♥ / ♥ !
/) / H K r/  / / H  ! / P  - / H "
445 A$(91,180)="  / / r/ H . + / h  / / / P  D
/ r/ H . + /  / J   H P r/ H ♥ + / P ♥ ♥ × / ↓"
450 A$(181,201)="+/ D hhHC Pp G H IL_d"
455 A$(172,172)=CHR$(155):A$(148,148)=CHR$(34):E$(49,49)=C
HR$(34):POKE 1536,96
480 REM CLEAR PAGE 6 & RESERVE AREA AT TOP OF MEMORY
490 GOSUB 130
500 A=A+1:E=PEEK(A*256):POKE A*256,128:IF PEEK(A*256)=128 TH
EN 500
510 POKE 106,A-17:GRAPHICS 0:POKE 559,0:A=(PEEK(106)+4)*256:
E=A:RETURN
680 TEMP=INT(V/256):POKE L,V-TEMP*256:POKE L+1,TEMP:RETURN
690 ? "K":? "▶I COULD NOT READ ";PLAY$
700 FOR X=1 TO 300:NEXT X:RUN
710 ? :? "KDISK FILE MUST HAVE .MBD EXTENTION":GOTO 700
720 ? :? :? " Just press RETURN to list MBD files"
730 POKE 752,1:POKE 201,8:? "K↓↓↓":CLOSE #3:OPEN #3,6,0,"D:*
.MBD"
740 INPUT #3,DR$:IF DR$(5,9)="FREE " THEN 760
750 ? ,DR$:GOTO 740
760 CLOSE #3:? :? "▶ PRESS ANY KEY TO CONTINUE";:POKE 764,25
5
770 IF PEEK(764)=255 AND PEEK(53279)=7 THEN 770
780 RUN
800 GRAPHICS 17:POKE 711,63:POKE 710,174
810 POSITION 4,5:? #6;"PRESS OPTION":POSITION 6,7:? #6;"for
menu"
820 POSITION 4,12:? #6;"PRESS START":POSITION 6,14:? #6;"to
rerun"
830 JW=0:POKE 53279,0:POKE 708,31:POKE 540,30
840 IF PEEK(53279)=3 THEN 900
850 IF PEEK(53279)=6 THEN RUN
860 IF PEEK(540)=0 THEN 880
870 GOTO 840
880 IF JW THEN 830
890 JW=1:POKE 53279,0:POKE 708,0:POKE 540,15:GOTO 840
900 GRAPHICS 0:POKE 752,1:? :? ," LOADING MENU":TRAP 910:RUN
 "D:MENU"
910 ? :? ,"KMENU NOT FOUND":POKE 540,120
920 IF PEEK(540)<>0 THEN 920
930 GOTO 800
```

The Playtest program plays *MusicBox* song data files using a machine language routine while BASIC is free to run another program. As supplied, the Playtest program asks for the name of the .MBD file to be played, reads this file from Disk Drive 1, then begins to play it during what is known as Vertical Blank.

If you don't remember the filename, press RETURN. This will give you a screen display of all files on Disk Drive 1 with the extention .MBD.

Once the music begins, you may press the OPTION key to stop the music. You may also press the START key to list the program. Notice that the music continues even as the program listing scrolls down the screen.

When the OPTION key is pressed, the music stops immediately and you are given two new options. Press the OPTION key again and you will return to the MENU program. Press the START key and you will restart the music.

If you look at the Playtest program listing, you will notice some REMark statements that provide information on how this program works. You may delete lines 351 through 500, then enter your own BASIC program starting at line 351 and ending at line 7998. Lines may be deleted and entered without disturbing the music. You can always stop the music by giving the immediate mode command GOTO 340:STOP. If you change the GOTO 400 command at the end of line 320 to a STOP command, you can restart the music by giving the immediate mode command GOTO 320.

Although line 320 actually starts or restarts the music, there are certain situations in which the sound registers must be initialized before restarting. This is done in line 310. To play it safe: when you add your own code to Playtest, always restart the music with a GOTO 310 command.

If you decide to alter this program, **stop the music** before saving your version. And, be sure to use a different program name!

Also: remember to **stop the music** before returning to the MENU program or doing any other disk or cassette I/O (Input/Output).

When adding your own routines, you could possibly cause problems by running into the song data in memory or destroying the machine language routines in memory. In the worst case, the computer will just "lock up." If the SYSTEM RESET key doesn't help you, reboot the computer.

```
0 GOSUB 8000:REM PLAYTEST by Jerry White [5/21/82]
1 REM
110 POKE 1536,96:X=USR(ADR(M$),ADR(E$)+115,1578,154)
120 FOR X=1537 TO 1577:POKE X,0:NEXT X:A=PEEK(106):RAMTOP=A
130 A=A+1:E=PEEK(A*256):POKE A*256,128:IF PEEK(A*256)=128 TH
EN 130
140 POKE 106,A-17:GRAPHICS 0:POKE 710,0:POKE 752,1:A=(PEEK(1
06)+4)*256:E=A
150 POSITION 11,11:? "READING ";PLAY$
200 RMUSIC=ADR("hh.Uh.T K♥ K♥ K♥ K♥ K♥IJp/ K♥┘AS K♥IJPI♦")

210 GBYTE=ADR("█+J\Q8┘J♥QH┘QI┘ Vdo █♥TIfTP fU♦JJ♦")
220 POKE 205,76:L=206:V=GBYTE:GOSUB 6000
230 FOR X=1 TO 4:A(X)=E:E=USR(RMUSIC,E):POKE E-2,0:POKE E-1,
0:NEXT X
300 CLOSE #1:POKE 710,160:? "K":POKE 752,1:? :? " YOUR SONG
IS NOW BEING PLAYED DURING"
302 ? :? "▸VERTICAL BLANK INTERRUPTS.":GOSUB 500:POKE 559,34


310 FOR X=0 TO 3:SOUND X,0,10,0:NEXT X:POKE 201,9
315 REM LINE 320 STARTS THE MUSIC
320 X=USR(ADR(E$),A,A(2),A(3),A(4)):GOTO 400
330 REM LINE 340 STOPS THE MUSIC
340 X=USR(ADR(E$)+92)
350 ? "K":? :? ,"PRESS OPTION FOR MENU":? :? ,"PRESS START T
O RESTART"
351 REM TO ENTER YOUR OWN BASIC CODE
352 REM  DELETE LINES 350 THRU 500
353 REM ENTER YOUR INSTRUCTIONS FROM
354 REM  LINE 350 UP TO LINE 7998.
355 FOR X=1 TO 100:NEXT X
360 IF PEEK(53279)=3 THEN POKE 106,RAMTOP:POKE 201,10:? "K":
? :? ," LOADING MENU":RUN "D:MENU"
370 IF PEEK(53279)=6 THEN 300
380 GOTO 360
400 IF PEEK(53279)=3 THEN 340
404 IF PEEK(53279)=6 THEN LIST :GOSUB 500
410 GOTO 400
500 POKE 752,1:? :? ,"PRESS OPTION TO END":? :? ,"PRESS STAR
T TO LIST":RETURN
6000 TEMP=INT(V/256):POKE L,V-TEMP*256:POKE L+1,TEMP:RETURN


7999 GOSUB 350:STOP
8000 GRAPHICS 0:POKE 559,0:POKE 710,160:POKE 82,1:POKE 83,38

:POKE 201,10:DIM DR$(20)
8010 DIM M$(88):M$="hI┘PQh.ah.Qh.Ch.bh.eh.d)♥.f.g(1Q.b┘)├eQ.
Q)♥ea.a┘)├eb.b)♥eC.C┘)├ef.f)♥eg.g┘%dE(PN%eEgPH♦"
8020 ? :? :? ,"MUSICBOX PLAYTEST":? :? :? " The input disk f
```

```
ile to this program"
8040 ? :? " must be a CONVERTED MUSIC COMPOSER"
8060 ? :? "     file with an .MBD extention.":? :? "       Dis
k drive 1 is assumed."
8080 ? :? :? " Just press RETURN to list MBD files"
8100 DIM A(4),E$(269),USER$(12),PLAY$(15):PLAY$="D1:":POKE 7
64,255:POKE 559,34
8120 ? :? " or enter disk filename";:POKE 53279,0:POKE 53279
,0:INPUT USER$:LU=LEN(USER$):IF LU=0 THEN 12000
8125 TRAP 10000:FOR ME=1 TO LU:IF USER$(ME,ME)="." THEN POP
:GOTO 8130
8126 NEXT ME
8127 IF LU>8 THEN ME=9:GOTO 8130
8128 USER$(LEN(USER$)+1)=".MBD":LU=LU+4:GOTO 8140
8130 USER$(ME,ME+3)=".MBD":LU=LEN(USER$)
8140 PLAY$(LEN(PLAY$)+1)=USER$:POKE 752,1
8200 TRAP 9000:CLOSE #1:OPEN #1,4,0,PLAY$:TRAP 40000
8300 E$(1,90)="[illegible]"
8310 E$(91,115)="[illegible]"
8320 E$(116,207)="[illegible]"
8330 E$(208,269)="[illegible]"
8340 E$(49,49)=CHR$(34):E$(240,240)=CHR$(34):RETURN
9000 ? :? "▶I COULD NOT READ ";PLAY$:FOR X=1 TO 300:NEXT X:R
UN
10000 ? :? "[illegible]  DISK FILE MUST HAVE .MBD EXTENTION":GOTO 8120


12000 POKE 752,1:POKE 201,8:? "[illegible]":CLOSE #3:OPEN #3,6,0,"D

:*.MBD"
12100 INPUT #3,DR$:IF DR$(5,9)="FREE " THEN 13000
12200 ? ,DR$:GOTO 12100
13000 CLOSE #3:? :? "▶ PRESS ANY KEY TO CONTINUE";:POKE 764,
255
13100 IF PEEK(764)=255 AND PEEK(53279)=7 THEN 13100
13200 RUN
```

Musicbox Program

This program is designed to help you understand how to transpose sheet music into the *Music Composer*.

The first display demonstrates the use of a redefined character set and displays a whole note, half notes, quarter notes, eighth notes, and sixteenth notes. Notice the duration of each sound. Press any key to continue.

The second display is used to create notes. For example, if you want middle C, use this display to put it on the screen. You do this by either moving the joystick or pressing one of the ARROW keys. (The joystick must be plugged into Jack No. 1).

Every note that can be entered into the *Music Composer* can be displayed on the screen. As this display changes, the appropriate musical sound is heard. Near the bottom of the screen you will see the pitch number and *Music Composer* interpretation of each note. This display does not include the timing value, such as quarter note or half note. This display was designed to show the note value of any note within *Music Composer's* range.

To make the note lower, pull back on the joystick or use the DOWN ARROW key. To make the note higher, push forward on the joystick or use the UP ARROW key. (When using the arrow keys, you do not have to hold the CONTROL key.)

Sharps and flats may also be displayed. However, in this program certain notes do not have sharps or flats. For example, C has a sharp but C-flat would actually be B-natural. B has a flat but B-sharp would be C-natural. F has a sharp but F-flat would be a natural E. And E has a flat but E-sharp would be a natural F.

To get the program to display a flat, move the joystick left or use the LEFT ARROW key. To produce a sharp, move the joystick right or use the RIGHT ARROW key. If the sharp or flat is not possible, as explained above, either no change will occur or the natural will be demonstrated. Also, no change will occur if you attempt to go higher or lower than the range of *Music Composer*.

To exit this program, press the joystick button or the ESCAPE key.

The Musicbox program makes use of "players" (graphics that are part of the ATARI Player/Missile Graphic System) for the sample note, treble and bass clefs, sharps, and flats.

```
0 GOTO 600:REM MUSICBOX Jerry White (5/21/82)
1 REM
11 KEY$(3,3)=KEY$(2,2):KEY$(2,2)="S":RETURN
12 KEY$(3,3)=KEY$(2,2):KEY$(2,2)="F":RETURN
19 SOUND 0,0,0,0:A=USR(ADR(ERASE$),PMBASE)
20 IF VP=100 AND S=13 THEN VP=112:GOTO 25
21 IF VP=100 AND S=14 THEN VP=96
25 RESTORE VP:READ KEY$,PITCH,FLT,SHP:KEY$(3,3)=" "
26 IF FLAT AND NOT FLT THEN FLAT=0
27 IF SHARP AND NOT SHP THEN SHARP=0
29 CIR$(2,2)=CHR$(VP):A=USR(ADR(PM$),ADR(CIR$),LEN(CIR$),PMB
ASE)
30 A=USR(ADR(PM$),ADR(TRC$),LEN(TRC$),PMBASE):A=USR(ADR(PM$)
```

```
,ADR(BAC$),LEN(BAC$),PMBASE)
31 IF SHARP THEN SHARP$(2,2)=CHR$(VP):A=USR(ADR(PM$),ADR(SHA
RP$),LEN(SHARP$),PMBASE):PITCH=SHP:GOSUB 11
32 IF FLAT THEN FLAT$(2,2)=CHR$(VP):A=USR(ADR(PM$),ADR(FLAT$
),LEN(FLAT$),PMBASE):PITCH=FLT:GOSUB 12
34 SOUND 0,PITCH,10,14:POSITION 15,19:? KEY$:POSITION 24,19:
? PITCH;" ";:SOUND 0,PITCH,10,2:RETURN
40 DATA C6,29,0,0
44 DATA B5,31,33,0
48 DATA A5,35,37,33
52 DATA G5,40,42,37
56 DATA F5,45,0,42
60 DATA E5,47,50,0
64 DATA D5,53,57,50
68 DATA C5,60,0,57
72 DATA B4,64,68,0
76 DATA A4,72,76,68
80 DATA G4,81,85,76
84 DATA F4,91,0,85
88 DATA E4,96,102,0
92 DATA D4,108,114,102
96 DATA C4,121,0,114
112 DATA C4,121,0,114
116 DATA B3,128,136,0
120 DATA A3,144,153,136
124 DATA G3,162,173,153
128 DATA F3,182,0,173
132 DATA E3,193,204,0
136 DATA D3,217,230,204
140 DATA C3,243,0,230
200 FOR ME=10 TO 100 STEP 10:POSITION X,Y:? M1$;:SOUND 0,0,0
,0:SOUND 0,ME,4,8:POSITION X,Y:? M2$;
210 POKE 755,1:SOUND 0,0,0,0:SOUND 0,ME,0,8:POSITION X,Y
220 ? M3$:SOUND 0,0,0,0:SOUND 0,ME,10,8:POSITION X,Y:? M4$:S
OUND 0,0,0,0:SOUND 0,ME,12,8:POKE 755,2:NEXT ME
230 JW=Y:IF JW>15 THEN JW=0
240 POSITION X,Y:? "c";:SOUND 0,0,0,0:SETCOLOR 2,JW,2:RETURN

300 FOR ME=1 TO T:FOR V=12 TO 0 STEP D:SOUND 0,0,2,V:NEXT V:

NEXT ME:SOUND 0,0,0,0:RETURN
800 GRAPHICS 0:POKE 559,0:POKE 82,1:DIM F$(32),MM$(82),KEY$(
10):F$="h[illegible]"
810 C5=PEEK(106)-20:F$(15,15)=CHR$(C5):CLOSE #3:OPEN #3,4,0,
"D:MUSIC.FNT":JW=USR(ADR(F$)):CLOSE #3:POKE 756,C5
815 DIM WNS$(1),HNUS$(12),HNDS$(5),SNUS$(12),SNDS$(12)
820 DIM QNUS$(12),QNDS$(5),ANUS$(12),ANDS$(5),B$(82)
830 B$="        ↓←←←←←←←        ↓←←←←←←←        ↓←←←←←←←
↓←←←←←←←        ↓←←←←←←←        "
840 DIM TRC$(90),BAC$(90),M1$(1),M2$(1),M3$(1),M4$(1),PM$(81
),ERASE$(24):M1$="g":M2$="h":M3$="i":M4$="j":POKE 710,0
841 TRC$="├?V♥├|||||├♥┘┐ ▪┌#EJ$┼▘┘ !┼▘♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥
♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥├ H◢┴▘▪ ♥♥♥♥♥♥♥♥♥├─┘//◢♥"
842 BAC$=" |?V[illegible]e ┼◢┼ e[illegible]♥♥e■┼◢H:◤[illegible]e[illegible]♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥
♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥♥[illegible]◆00┴┴▘▪ ┴┴00◆◆[illegible]♥♥♥♥♥"
843 DIM SHARP$(12),FLAT$(12),CIR$(11):CIR$="♥(\♥┴<<<┴♥♥":CIR
```

```
$(2,2)=CHR$(60):VP=60
844 PMADD=PEEK(106)-16:POKE 559,62:POKE 54279,PMADD:POKE 704
,31:PMBASE=(PMADD+4)*256
845 SHARP$="┘ (◢$$▯$$▯$$♥"
846 ERASE$="hh▯Mh▯L▯H ▯♥▯♥▯LHP▯FMJPV◆"
847 PM$="hh▯Lh▯Khh▯Mh▯Nh▯♥▯♥LK ▯♥/HDMPV┴▯♥/e▯N▯Ph▯♥/▯Oh▯♥/▯Oh
♥▯♥/▯OhHFQXOI♥Pq▯♥/I♥p/h ♥┴▯N◆"
848 FLAT$="┘ (◢@@@XdDX◆♥"
849 POKE 53248,0:POKE 53249,0:POKE 53250,0:POKE 53251,0:POKE
 705,159:POKF 706,159:POKE 707,207
850 DIM L$(20):L$="────────────────────"
932 WNS$="├":HNUS$=" ↓← ↑/↓← |":HNDS$="↓┘↓← ▀"
933 QNUS$=" ↓← ↑/↓←┤":QNDS$="↓┐↓← ▀"
934 ANUS$=" ↓← ↑/\↓←←┤":SNUS$=" ↓← ↑/◢↓←←┤":ANDS$="↓┐↓← ▄":SN
DS$="↓┐↓←◣"
2000 ? "K":POKE 710,160:POKE 752,1:POSITION 4,1:GOSUB 9999:?
 MM$
2005 FOR Y=7 TO 15 STEP 4:POSITION 3,Y:? "#":NEXT Y
2010 GOSUB 8000:POSITION 3,4:? " #";:GOSUB 8000
2020 POSITION 4,4:GOSUB 8000:? "c#":GOSUB 8000:POSITION 5,4:
? "c#":GOSUB 8000:POSITION 6,4:? "c";
2030 X=7:Y=4:GOSUB 200
2040 FOR X=10 TO 18:POSITION X,4:? " ";WNS$;:NEXT X:? "  WHO
LENOTE":T=1:D=-0.1:GOSUB 300
2050 POSITION 4,1:? B$:GOSUB 9998:POSITION 4,4:? MM$
2110 GOSUB 8000:POSITION 3,7:? " #";:GOSUB 8000
2120 POSITION 4,7:GOSUB 8000:? "c#":GOSUB 8000:POSITION 5,7:
? "c#":GOSUB 8000:POSITION 6,7:? "c";
2130 X=7:Y=7:GOSUB 200
2140 FOR X=10 TO 18:POSITION X,7:? HNUS$;:NEXT X:? "  HALFNO
TES":POSITION 32,7:? HNDS$:T=2:D=-0.2:GOSUB 300
2150 POSITION 4,4:? B$:GOSUB 9998:POSITION 4,8:? MM$
2210 GOSUB 8000:POSITION 3,11:? " #";:GOSUB 8000
2220 POSITION 4,11:GOSUB 8000:? "c#":GOSUB 8000:POSITION 5,1
1:? "c#":GOSUB 8000:POSITION 6,11:? "c";
2230 X=7:Y=11:GOSUB 200
2240 FOR X=10 TO 18:POSITION X,11:? QNUS$;:NEXT X:? "  QUART
ERNOTES":POSITION 35,11:? QNDS$:D=-0.4:GOSUB 300
2250 POSITION 4,8:? B$:GOSUB 9998:POSITION 4,12:? MM$
2310 GOSUB 8000:POSITION 3,15:? " #";:GOSUB 8000
2320 POSITION 4,15:GOSUB 8000:? "c#":GOSUB 8000:POSITION 5,1
5:? "c#":GOSUB 8000:POSITION 6,15:? "c";
2330 X=7:Y=15:GOSUB 200
2340 FOR X=10 TO 18:POSITION X,15:? ANUS$;:NEXT X:? "  EIGHT
HNOTES":POSITION 34,15:? ANDS$:D=-0.8:GOSUB 300
2350 POSITION 4,12:? B$:GOSUB 9998:POSITION 4,16:? MM$
2410 GOSUB 8000:POSITION 3,19:? " #";:GOSUB 8000
2420 POSITION 4,19:GOSUB 8000:? "c#":GOSUB 8000:POSITION 5,1
9:? "c#":GOSUB 8000:POSITION 6,19:? "c";
2430 X=7:Y=19:GOSUB 200
2440 FOR X=10 TO 18:POSITION X,19:? SNUS$;:NEXT X:? "  16TH
NOTES":POSITION 34,19:? SNDS$:D=-1.6:GOSUB 300
2450 POSITION 4,16:? B$:POSITION 3,4:? "THIS NOTE";:? :? :?
:? "DURATION DEMO";:? :? :? :? "USES REDEFINED";
2460 ? :? :? :? "CHARACTER SET";:? :? :? :? "  GRAPHICS.";:?
```

```
 :? :? :? "■=SHARP ¯=FLAT";
2500 FOR X=1 TO 50:NEXT X:FOR X=245 TO 0 STEP -2:POKE 712,X:
SOUND 0,X,10,2:SOUND 1,X+2,10,2:SOUND 2,X+4,10,2
2510 SOUND 3,X+6,10,2:NEXT X:SOUND 0,0,0,0:SOUND 1,0,0,0:SOU
ND 2,0,0,0:SOUND 3,0,0,0
2550 POSITION 7,23:? "PRESS ANY KEY TO CONTINUE";:POKE 764,2
55
3000 POKE 755,2:IF PEEK(53279)=7 AND PEEK(764)=255 THEN POKE
 755,1:GOTO 3000
3100 POKE 755,2:POKE 764,255
4000 ? "ß":POKE 752,1:POKE 710,16:POKE 709,8:POKE 53248,124:
POSITION 19,1:? "——":POSITION 19,2:? "——"
4010 POKE 53277,3:A=USR(ADR(ERASE$),PMBASE):POKE 53249,90:PO
KE 53250,98:POKE 53251,116
4100 FOR Y=3 TO 7:POSITION 10,Y:? L$:NEXT Y:POKE 766,1:POSIT
ION 19,8:? "——"
4120 POSITION 10,9:POKE 766,1:? "↑ TREBLE       ↓ BASS":POKE
766,0:POSITION 19,10:? "——"
4200 FOR Y=11 TO 15:POSITION 10,Y:? L$:NEXT Y:VP=40:CIR$(2,2
)=CHR$(VP):GOSUB 19
4220 POSITION 11,18:? "M.I.C.NOTE    PITCH":POSITION 9,21:? "
ESC OR TRIGGER TO QUIT"
4300 POKE 764,255:SHARP=0:FLAT=0
4301 K=PEEK(764):IF K=255 THEN 4310
4302 IF K=15 OR K=143 THEN POKE 764,255:S=13:GOTO 4450
4303 IF K=14 OR K=142 THEN POKE 764,255:S=14:GOTO 4400
4304 IF K=28 THEN 5000
4305 IF K=6 OR K=134 THEN S=11:GOTO 4410
4306 IF K=7 OR K=135 THEN S=7:GOTO 4460
4309 GOTO 4300
4310 S=STICK(0):TR=STRIG(0):IF  NOT TR THEN 5000
4320 IF S=15 THEN 4301
4400 IF S=14 AND VP>41 THEN VP=VP-4:GOSUB 19
4410 IF S=11 THEN FLAT=1:SHARP=0:GOSUB 19
4450 IF S=13 AND VP<139 THEN VP=VP+4:GOSUB 19
4460 IF S=7 THEN FLAT=0:SHARP=1:GOSUB 19
4500 GOTO 4300
5000 POKE 764,255:FOR X=53248 TO 53251:POKE X,0:NEXT X:POKE
53277,0:SOUND 0,0,0,0:POKE 756,224
5100 ? "ß":? :? ,"   LOADING MENU":RUN "D:MENU"
8000 POKE 540,6
8001 IF PEEK(540)<>0 THEN 8001
8002 RETURN
9998 MM$="        ↓++++++++cccccccc↓++++++++cMUSICc↓++++++++ cccc
cc↓++++++++ccBOXcc↓++++++++cccccccc":RETURN
9999 MM$="        ↓++++++++cccccccc↓++++++++cMUSICc↓++++++++■cccc
cc↓++++++++ccBOXcc↓++++++++cccccccc":RETURN
```

Translat Program

This program translates *Music Composer* files into the format used when entering data. This comes in quite handy when you are trying to solve a timing problem or when you are trying to find that "sour" note you entered by mistake.

The program was donated by Charles Bachand and *A.N.A.L.O.G. 400/800* Magazine, and altered to conform to *MusicBox* standards.

Music Composer files may be "dumped" either onto the screen or onto a printer. (Note: any compatible 40 or 80 column printer may be used.) The program will ask you to type P if you want a printed copy or S for Screen Display only. Since this response is only a single keystroke, you do not have to press the RETURN key.

You will then be asked to enter the *Music Composer* filename for the file you wish to have displayed or printed. Only enter the names of *Music Composer* files. This program will NOT read converted *MusicBox* files. If you forget the .MUS extention, the program will append it for you. If you have used an extention other than .MUS, the program will change it for you.

Once the file has been printed or displayed, press the OPTION key to rerun the program. Or, press the START key to return to the MENU program.

```
0 REM TRANSLAT 5/22/82
100 REM *** MUSIC DECOMPOSER ***
110 REM COPYRIGHT 1981 C.BACHAND
120 REM DECOMPOS MODIFIED BY JERRY WHITE

130 DIM F$(15),C$(1),V$(14),N$(7),D$(6),U$(12)
140 GOSUB 800:V$="0 PPP MPMFF FF":N$="CDEFGAB":D$="TSEQHW"

200 GET #1,A:IF A=255 THEN PRINT #P:PRINT #P;"END OF FILE":G
OTO 1500
210 IF A<>170 THEN GOTO 200
220 PRINT #P:GET #1,A:IF A<128 THEN 300
230 PRINT #P;"MISCELLANEOUS":PRINT #P
240 GET #1,A:GET #1,B:PRINT #P;"METER=",B;"/";A
250 GET #1,A:PRINT #P;"TEMPO=",A
260 GET #1,A:C$="S"
270 IF A>127 THEN A=A-128:C$="F"
280 PRINT #P;"KEY=",A;C$:GET #1,A:GOTO 200
300 IF A<20 THEN 500
310 PRINT #P;"ARRANGE VOICE #";A/2-9:PRINT #P:L=1
320 GET #1,A:IF A=255 THEN 200
```

```
330 PRINT #P;"#";L,:L=L+1:GET #1,B
340 ON A+1 GOTO 345,350,360,370,380,390,400
345 PRINT #P:GOTO 320
350 PRINT #P;"GOTO LINE ";B:GOTO 320
360 PRINT #P;"PLAY PHRASE ";B:GOTO 320
370 PRINT #P;"TRANSPOSE ";
372 IF B>128 THEN PRINT #P;"-";:B=B-128
375 PRINT #P;B:GOTO 320
380 PRINT #P;"VOLUME ";V$(B+B+1,B+B+2):GOTO 320
390 PRINT #P;"DISPLAY":GOTO 320
400 PRINT #P;"COUNT ";:GOTO 372
500 PRINT #P;"PHRASE  #";A/2:M=1
510 GET #1,A:IF A=255 THEN 200
520 GET #1,B:IF A=127 THEN PRINT #P:PRINT #P;"MEASURE #";M:P
RINT #P:M=M+1:N=1:GOTO 510
530 PRINT #P;"#";N,:N=N+1:B=B-1
540 IF A=85 THEN PRINT #P;"R";:GOTO 570
550 C$="":A=A/4:L=A-INT(A):IF L THEN C$="S":A=INT(A):IF L=0.
5 THEN C$="F"
560 L=A-INT(A/7)*7+1:PRINT #P;N$(L,L);C$;INT(A/7)+3;
570 C$="":IF B>127 THEN C$="T":B=B-128
580 F$="":L=INT(B/2)+1:IF B-INT(B/2)*2 THEN F$="."
590 PRINT #P;D$(L,L);F$;C$:GOTO 510
600 GRAPHICS 2:PRINT #6;"     ##################":PRINT #6;"   #### M
usic ####":PRINT #6;"  # translator. #"
610 PRINT #6;" ##############################":PRINT #6;" A ANALOG 400/80
0 A":PRINT #6;" *** Magazine ***"
620 ? #6;"    (617) 892-3408":? #6;"      $12. annual":? #6;"
   subscription":POKE 82,0:DIM R$(20)
660 PRINT "R TYPE P FOR PRINTER OR S FOR SCREEN";:P=0:POKE 7
64,255
670 JW=PEEK(764):IF JW=255 THEN 670
672 IF JW=10 THEN P=2:CLOSE #P:OPEN #P,8,0,"P:":GOTO 900
674 IF JW=62 THEN P=0:GOTO 900
676 ? "R":GOTO 660
900 POKE 764,255:? :? :? "}DISK DRIVE 1 IS ASSUMED."
940 PRINT "   ENTER MUSIC FILE NAME";:INPUT U$:LU=LEN(U$):IF
 LU=0 THEN GOSUB 32100:GOTO 900
952 F$="D:":LU=LEN(U$):TRAP 1000:FOR ME=1 TO LU:IF U$(ME,ME)
=".." THEN POP :GOTO 950
953 NEXT ME
954 IF LU>8 THEN ME=9:GOTO 950
956 U$(LEN(U$)+1)=".MUS":LU=LU+4:GOTO 959
958 U$(ME,ME+3)=".MUS":LU=LEN(U$)
959 F$(LEN(F$)+1)=U$
960 TRAP 2000:POKE 82,2:CLOSE #1:OPEN #1,4,0,F$:TRAP 40000
965 GRAPHICS 0:POKE 710,0:? :? "WORKING":? :PRINT #P;F$
970 RETURN
1000 ? "RDISK FILES MUST HAVE .MUS EXTENTION":GOSUB 3000:GO
TO 900
1500 POKE 752,1:? "R":? :? ,"PRESS OPTION TO RERUN":? ,"PRES
S START FOR MENU"
1510 IF PEEK(53279)=3 THEN CLOSE #1:CLOSE #2:RUN
1520 IF PEEK(53279)=6 THEN CLOSE #1:CLOSE #2:RUN "D:MENU"
```

```
1530 GOTO 1510
2000 ? "K█I COULD NOT OPEN ";F$:GOSUB 3000:RUN
3000 FOR JW=1 TO 500:NEXT JW:RETURN
32100 CLOSE #3:OPEN #3,6,0,"D:*.MUS":GOTO 32300
32300 GRAPHICS 0:POKE 710,0:POKE 752,1:TRAP 32400:? :? ,"FIL
ES AVAILABLE":?
32310 INPUT #3,R$:? ,R$:GOTO 32310
32400 ? :? "▸PRESS ANY KEY TO CONTINUE":POKE 764,255
32500 IF PEEK(764)=255 AND PEEK(53279)=7 THEN 32500
32600 ? :? :POKE 764,255:POKE 752,0:RETURN
```

ATARI Basic Chords

CHORD	DIST	FIRST	THIRD	FIFTH	DIST	FIRST	THIRD	FIFTH
A Major	10=	144	114	96	12=	75	60	51
A Minor	10=	144	121	96	12=	75	63	51
A# or Bb Major	10=	136	108	91	12=	72	57	48
A# or Bb Minor	10=	136	114	91	12=	72	60	48
B Major	10=	128	102	85	12=	67	55	45
B Minor	10=	128	108	85	12=	67	57	45
C Major	10=	121	96	81	12=	63	51	42
C Minor	10=	121	102	81	12=	63	55	42
C# or Db Major	10=	114	91	76	12=	60	48	40
C# or Db Minor	10=	114	96	76	12=	60	51	40
D Major	10=	108	85	72	12=	57	45	37
D Minor	10=	108	91	72	12=	57	48	37
D# or Eb Major	10=	102	81	68	12=	55	42	36
D# or Eb Minor	10=	102	85	68	12=	55	45	36
E Major	10=	96	76	64	12=	51	40	33
E Minor	10=	96	81	64	12=	51	42	33
F Major	10=	91	72	60	12=	48	37	31
F Minor	10=	91	76	60	12=	48	40	31
F# or Gb Major	10=	85	68	57	12=	45	36	30
F# or Gb Minor	10=	85	72	57	12=	45	37	30
G Major	10=	81	64	53	12=	42	33	28
G Minor	10=	81	68	53	12=	42	36	28
G# or Ab Major	10=	76	60	50	12=	40	31	27
G# or Ab Minor	10=	76	64	50	12=	40	33	27