Zero Free

Pack every sector on your disks for efficient storage.

by Mike Stortz

Over the years, I’ve collected quite a few binary load programs, distributed over thirty-two disk sides. I’ve always tried to fit these programs together on a disk as economically as possible, spending hours over a hot calculator, but there were usually ten or twenty free sectors left over.

I recently obtained a "boot menu," a program that eliminated the need to have DOS and an AUTORUN.SYS file on each disk. I had to redistribute my programs to take advantage of this extra space anyway, so I decided that a utility program to replace my calculator—and callused digits—was in order. Zero Free was the result.

Since some amount of speed was obviously going to be needed, I naturally chose to write in Action! I take second place to few people in my admiration of this excellent language. Machine language is fine where speed or small size are necessities rather than conveniences, but it’s a terror to debug (at least, it’s a terror for me to debug). I knew that I wanted to combine file lengths to add up to 707 sectors, but I had no idea how to go about it. I wrote several “intelligent” (and unsuccessful) algorithms before deciding to do the job with brute computing power.

Zero Free will read in the directories of your disks (ignoring any .SYS extenders) and fiercely recombine them randomly, until it meets with a favorable arrangement. You may be surprised at how few programs are required for Zero Free to come up with a completely full disk. It then prints out the appropriate filenames to a disk file, the screen or your printer.

Using Zero Free.

Insert your Action! cartridge, and type and save Listing 1. Please use D:Check in Action! from issue 44 to check your typing.

When run, Zero Free will provide you with initial instructions. Give the number of free sectors you have per disk, the maximum number of files you wish to have on a disk side (I did this because my labeling program will only fit seven filespecs on a label), and D, S or P, depending on whether you wish the program output to go to a file named D:PRINTOUT, the screen or your printer, respectively.

Insert each disk that has files you want to pack and press the SPACE BAR. The directory of each disk will be displayed. If a program on the current disk has the same name as a file previously entered, a number sign (#) appears next to it. If these two files are also of equal length, an equal sign (=) appears. You’ll probably want to eliminate duplicate files. You can do this by pressing the letter next to the unwanted program. You can also add an extra file, by pressing the plus sign (+) and giving the filename and its length.

When you’ve finished entering files, press ESC to quit data entry and begin calculation. Zero Free will produce a list of files that will fill as much space as possible on a disk side. If you’ve selected output for D:PRINTOUT, you will be prompted for a disk to write it on.

When zero waste is no longer possible, Zero Free will go for the minimum waste it can find. This process continues until all files have been assigned, or till the user has pressed a key and aborted the program.

Using Zero Free and a boot menu program like NoDOS, QuikLoad, or BOOT, I reduced my library from thirty-two to twenty disk sides, a savings of six disks. Now, I can put write-protect tabs and neat labels on all those disks, secure in the knowledge that I have—Zero Free sectors.

Mike Stortz is the P.D. Librarian for G.R.A.S.P., the Richmond, Virginia Atari users’ group. Seemingly unable to find employment in the programming field, he’s working on about thirty projects at once, including a graphic arcade/adventure game that will make Ultima III look like “Hunt the Wumpus.”
Listing 1.
Action listing.

; CHECKSUM DATA
; [E9 CA B3 3A 88 5A 82
; 43 F8 3F 9D 21 E0 1F F6
; 95 99 78 70 96 0E 4C 37
; 46 73 47 6D 69 38 F1 BC
; 37 AF 66 BC 79 89 7D
; D1 58 0C 7E 07 4C 3F 1]

BYTE btemp, spaces, len, checkflag,
maxfiles, devc, num, quit,
Imargin=82, shflk=702, ch=764
attract=77, crsinh=752, errno=75,
brkey=17

CARD idx, which, ctemp, sum, max, spare,
free, a, b, loss, lastloss, waste,
addlen, imperr, dl=588, sc=88

INT ii

BYTE ARRAY names(600), name(20),
extender(51), hold(324),
string1(14), string2(14)

CARD ARRAY length(500), hlen(27),
programs(500), pr(500)

CARD FUNC MintCARD aa, bb)

IF aa(bb THEN RETURN(aa)
ELSE RETURN(bb)

FIX

PR QC ClearOut()

Position(2,17)
FOR a2i to 10 DO
Put(156)
OD
Position(2,18)

RETURN

PR QC MyError()

ClearOut()

IF brkkey=0 THEN
Error=ImpErr
Break
ELSE IF errn=159 THEN
Print("Disk Error 8")
PrintE(erno)
PutE(2)
Print("Check the drive and ")
Print("press a key.")
i=GetD(2)
ELSE
PrintE("Unexpected error.")
Print("Check things and press ")
Print("a key.")
i=GetD(2)

FIX

RETURN

PROC Title()

Imargin=8
Graphics(8)

FOR btemp=1 to 10 DO
Put(127) Put(158)
OD
Print(""
Put(159) Put(125)
Poke(d+i+9,7)
Poke(d+i+10,6)
Poke(710,194)
Poke(700,190)
Poke(712,192)
crsinh=1

Print("")
Print("Written in ACTION! by ")
Print("Mike Stortz ")
Print("G.R.A.S.P. of ")
Print("Richmond, Va.")
Print(""
Print("Zero")
Imargin=2
Print("<zero>(free) ")
Print("NO EMPTY SECTORS ")
Print("This program reads in ")
Print("the contents")
Print("of your binary file disks ")
Print("remembers")
Print("their lengths, and sorts ")
Print("them to")
Print("occupy the least number")
Print("of disk space")
Print("ZEROFREE will hold about")
Print("of disk space")
Print("programs & their lengths.")

PutE(2)
Print("A disk has 787 free")
Print("sectors if you")
Print("use a boot menu like ")
Print("QuickLoad, or 660")
Print("sectors minus the length")
Print("of your menu")
Print("if using DOS.")

PutE(2)
Print("A ""
Print("before a filename")
Print("if it is a duplicate, ")
Print("or a "")
Print("appear if it is of ")
Print("equal length.")

PutE(2)
Print("** Please press ")
Print("a key **")
i=GetD(2)
crsinh=0
free=0

DO 0 ClearOut()
Print("How many free sectors ")
Print("available?")
free=InputO()
UNTIL free>0 OD

maxfiles=0
DO 0 ClearOut()
Print("Maximum files per disk: ")
maxfiles=InputO()
UNTIL maxfiles>0 OD

devc=0

DO 0 ClearOut()
Print("Output to D:\PRINTOUT."
Print("screen.")
Print("printer")
Print("D/F/C? ")
devc=GetD(2)  
UNTIL devc='B' OR  
devc='P' OR  
devc='S' OD  
Graphics(0)  
Poke(710,194)  
crsin=1  
RETURN

PROC GetDir()  
Put(125)  
ClearOut()  
Print("Now up to ")  
PrintC(Max3)  
Print(" programs.")
num=0  
Close(1)  
Open(1,"D:\*,.W",6,0)  
DO  
InputMDB(1,name,18)  
MoveBlock(extend+1,name+11,3)  
extender(0)=3  
i=5Compare(extend,"SYS")  
IF name(0)16 AND i=0 THEN  
num+=1  
MoveBlock(hold+num*12+i,name+3,  
11)  
hlen(num)=ValC(name+14)  
IF num>26 THEN EXIT FI  
FI  
UNTIL EOF(1) OD  
Close(1)  
RETURN

PROC PrintDir()  
BYTE dup  
Put(125)  
IF num=0 THEN  
Print(""")  
FOR btemp=1 TO MAX DO  
IF MAX=0 THEN  
FOR ctemp=1 TO MAX DO  
MoveBlock(string1+,  
hold+12Wbtemp+1,11)  
string1(0)=11  
MoveBlock(string2+,  
names+12Wbtemp+1,11)  
string2(0)=11  
i=5Compare(string1,string2)  
IF i=0 AND  
hlen(ctemp)=length(ctemp)  
THEN  
i+=10  
FI  
IF  
ii=0 OR ii=10 THEN EXIT FI  
OD  
FI  
IF ii=0 THEN  
dup=18  
ELSEIF ii=10 THEN  
dup=32  
ELSE  
dup=32  
FI  
PrintC("KX -KX5XC ",192+btemp,  
dup,hold+12Wbtemp,127)  
OD  
Pute()  
RETURN  

PROC CopyDir()  
MoveBlock(names+12+max*12,hold+12,  
num*12)  
MoveBlock(length+2+max*2,hlen+2,  
num*2)  
max+=num  
RETURN

PROC Add()  
ClearOut()  
SetBlock(string1,14,32)  
Print("Enter filename to add")  
PrintC("(No ' ' please)")  
InputMDB(0,string1,11)  
IF string1(0)=0 THEN RETURN FI  
string1(0)=32  
string1(0)=11  
ClearOut()  
Print("Enter length of ")  
PrintC(string1)  
addlen=InputC()  
IF addlen=0 OR addlen>400 THEN  
RETURN  
FI  
num+=1  
MoveBlock(hold+num*12,string1,12)  
hlen(num)=addlen  
RETURN

PROC Delete()  
btemp=-64  
IF btemp=num THEN  
MoveBlock(hold+btemp*12,  
hold+12Wbtemp+1,11)  
MoveBlock(string1+,  
hold+12Wbtemp+1,11)  
string1(0)=11  
MoveBlock(string2+,  
names+12Wbtemp+1,11)  
string2(0)=11  
nnum=1  
IF num=0 THEN  
nnum=-1  
FI  
RETURN  

PROC GetLibrary()  
DO  
IF idx>400 THEN EXIT FI  
PrintDir()  
ClearOut()  
Print("Insert next disk to ")  
Print("be cataloged")  
Print("(and press Chapter,)")  
Print("to delete,")  
Print(" to add, or")  
Print(" to quit & print")  
btemp=GetD(2)  
IF btemp=32 THEN  
RETURN

ANALOG COMPUTING
MAY 1987 / PAGE 33
Zero Free continued

```
COPYREG
GETREG
ELSE IF btemp=64 and btemp(64+num)
GETREG
ELSE btemp=8 THEN
GETREG
ELSEIF btemp=4 THEN
COPYREG
RETURN
DO
RETURN
PROC PrintName()
PRINT(1, names"which=MI5")
PRINT(1,"")
space=-1
IF space<0 OR (space=0 AND devc=0) THEN
space=-1
PUT(T1)
RETURN
PROC KeyCheck()
IF ch(255 THEN
RETURN
Put(t)
FOR idx=1 TO max DO
PRINT(t, names+"programs(idx)"+mi)
PRINT(1,"")
PrintCDE(1, length(programs(idx)))
DO
PRINT("Press ENTER.");
lj=Get(1)
QUIT
RETURN
PROC PrintIndex()
Put(t)
Put(t)
PRINT("#SUKKE", "programs left - ", max)
PRINT("#SUKKE", "sections wasted - ", spare)
PRINT("sections out...")
-PRINT("allowable waste - ", waste)
-PRINT("Press any key to abort")
DO
PRINT("This many free sectors:")
RETURN
PROC Switch()
IF btemp=0 THEN
GETREG
ENDSWITCH
ENDSWITCH
PROC PrintLibrary()
FOR idx=1 TO max DO
PRINT("programs(idx)=programs(which)"
-PRINT("program which=temps"
-RETURN
PROC PrintLibrary()
FOR idx=1 TO max DO
PRINT("programs(idx)=programs(whic"
-PRINT("program which=temps"
-RETURN
```