

# DISK UTILITIES 1.10.00

1/28/94

INDEX SECTORS = 00000008  
 E CAT. AREA = 00001279  
 C. RENT END = 00001073

NAME	TYPE	START	END	USED	FREE
@SCSICFG	F	00000008	00000090	00000075	00000008
@SYSMVFB	F	00000091	00000098	00000006	00000002
START	F	00000099	00000101	00000003	00000000
@MOVE1	F	00000102	00000141	00000039	00000001
@.BACKUP	F	00000142	00000147	00000004	00000002
@TD.DISK	F	00000148	00000208	00000061	00000000
@RAMDISK	F	00000209	00000223	00000015	00000000
@DSAPPLY	F	00000224	00000230	00000007	00000000
@MENU	F	00000231	00000269	00000036	00000003
@FORMAT	F	00000270	00000334	00000065	00000000
@.RESTOR	F	00000335	00000340	00000004	00000002
@.DISK	F	00000341	00000348	00000005	00000003
@TD.CREO	F	00000349	00000354	00000006	00000000
@BACKUP	F	00000355	00000409	00000055	00000000
@VERCPUC	F	00000410	00000412	00000003	00000000
@TDIMAGE	F	00000413	00000438	00000024	00000002
@DSCFIGP	F	00000439	00000442	00000003	00000001
@DSCFIG	F	00000443	00000555	00000110	00000003
@HITRATE	F	00000556	00000573	00000017	00000001
.STARTD	D	00000574	00000576	00000003	00000000
@MOVEFIL	F	00000577	00000645	00000067	00000002
@TD.CREF	F	00000646	00000703	00000056	00000002
@TAPEB	F	00000704	00000775	00000054	00000018
@TAPER	F	00000776	00000828	00000043	00000010
@RECOVER	F	00000829	00000882	00000054	00000000
@DSTAPEB	F	00000883	00000966	00000084	00000000
@VERCPUR	F	00000967	00000969	00000003	00000000
@VERCPUB	F	00000970	00000972	00000003	00000000
@DSTAPER	F	00000973	00001032	00000058	00000002
@TD.SUBS	F	00001033	00001073	00000041	00000000

114 USED AS OF 2/2/94

# DS/SCSI UTILITIES

- START
- CMENU
- .STARTD
- CSYSAVPB

MOVING A SELECTED LIST OF FILES FROM A DISK INDEX  
 MAIN MENU DISK MANAGEMENT UTILITIES 1/20/94 Disk Util 1.1 1994

- CFORMAT
- CMOVEFIL

ADD LINE 617 TO INSURE EITHER 2200 OR DOS PROPERLY SELECTED. 617 IF AZE="2" THEN 620:IF AZE="1" THEN 620:GOTO 615  
 1/27/93 SPLIT LINE 290 AFTER COMMAND W/ GOSUB MAKING LINE 300 TO ALLOW CONVERSION TO NEW.

- CBACKUP
- CBACKUP
- CVERCPUB
- CSCTAPEB
- CDSTAPEB

BACKUP UTIL

LVP BACKUP TO FLOPPY

220 APPEND NOT SUPPORTED. / ERASE TAPE (Y OR N):

APPEND ONTO TAPE OR ERASE TAPE (A OR E)  
~~ASKS TO APPEND Y/N OR ERASE Y/N~~

~~11/23 CHANGED OR DELETED THE WORD PLATTER TO ADD~~

- RESTOR
- CVERCPUR
- CSCTAPER
- CDSTAPER
- CRECOVER

RESTORE UTIL

- CTO.CRE0
- CTO.CREF
- CTO.IMAGE
- CTO.SUBS
- CTO.DISK

INITIAL LOAD PRGB

MAIN PART OF CREATING LIST  
 MOVING SELECTED FILES FROM A DISK INDEX

DISK MANAGEMENT UTIL MENU

11/15/92 CHANGED "SURFACE" TO "ADDRESSES" FOR PROTECT/UNPROTECT 1/20/94 CHANGED REL TO 1.10.  
 10/23/92 REMOVED PERIODS FROM DS, ADDED ? TO QUESTIONS. 4030, 4132, 4160, 4215, 8515 12/2/92  
 12/18/92 CHANGED REFERENCES TO "SURFACE" IN ALL SCREENS TO "ADDRESSES". 7/27/93

~~12/11/92 CHANGES~~

- CRAMDISK
- CVERCPUC
- CSCSI.CFG
- CDSAPPLY

11/19/92 2/16/93  
 LEFT OFF DISK UTILITIES.

## UTILITY CHANGES IN CHRONOLOGICAL ORDER

- 11/6/92 CGENPART - UNABLE TO EXECUTE W/ SF 15 IF CURRENT CONFIG > 8M.
- 11/15/92 CDSCFIG - CHANGED ALL REFERENCES TO 'SURFACE' TO 'ADDRESS' FOR PROTECT/UNPROTECT.
- 11/17/92 CPSTAT - ADDED MESSAGE ON SCREEN, 'SF 12/NEXT' + 'SF 13/REV' FOR SUBSEQUENT SCREENS
- 11/19/92 CSCSICFG - CHANGED SECTOR PROMPT MESSAGE. 2. ADDED TEST TO DETERMINE IF SECTOR SIZE ENTERED EXCEEDS AVAILABLE SECTORS. CORRECTED ERROR MESSAGE FOR THIS. 3. ADDED LINE 1280 TO CLEAR ERROR MESSAGE. 4. RESET SECTORS AVAILABLE TO FULL AVAILABILITY IF ENTER NO TO CONFIG ACCEPTABLE. 5. CHANGE LINE 1230 TO READ "# OF ADDRESSES TO ASSIGN".
- 11/23/92 CDSCFIG - REMOVED . & ADDED ? TO INPUTS ON LINES 4030, 4132, 4160, 4215, 8515
- 12/2/92 CDSCFIG - WHEN RESPONDED NO TO APPLY CHANGES CURSOR IS RETURNED. CORRECTED TO SEND BACK TO THE BASE ADDR. SCREEN. ADDED WARNING, LINES 8150 & 8155, WHEN APPLYING CHANGE.
- 12/3/92 CDSCFIG - ADDED WARNING TO MAKE BACKUP WHEN KEY SF 10 TO APPLY CONFIG. CHANGED MESSAGE FOR APPLY PROCEDURE COMPLETED, LINE 9010.
- 12/4/92 CSCSICFG - ADDED FN/TAB-EXIT TO ALL SCREENS, 1020, 1165, 1225, & 1440. ADDED WARNING TO APPLY SCREEN, 1440.
- 12/11/92 CDSCFIG - CHANGED ALL REFERENCES TO 'SURFACE' TO 'ADDRESS'.
- 12/15/92 CDSCFIG - CLEANED UP SCREENS FOR PASSWORD, CONFIG FILENAME, & ASSIGNMENTS; DS DEFAULTS DID NOT WORK, CORRECTED.
- 12/17/92 CSCSICFG - CLEARED ALL SECTORS COUNTS FROM ADDRESS IF RESPONDED 'No' TO CONFIGURATION ACCEPTABLE?. MADE ENTRIES < 100 UNACCEPTIBLE FOR SECTOR SIZE.
- 12/22/92 CSCSICFG - REMOVED 'Y' DEFAULT FROM ALL ENTRIES ACCEPTABLE?. EXTENDED MESSAGE FOR CONFIGURATION FILENAME.
- 12/23/92 CTO.CREP - CHANGED SCREEN HEADING.
- 1/15/93 CDSCFIG - ADDED NOTE TO INDICATE DS LEVEL NEEDED IF > 65535 SECTORS. ADDED LODS TO PREVENT OVERWRITE OF CONFIG FILE WITHOUT NOTICE.
- 1/22/93 CMOVEFIL - SPLIT LINE 290 TO ALLOW TRANSPARENT CONVERSION TO NEW FORMAT.

- 2/1/93 CFORMAT - ADDED LINE 617 TO INSURE EITHER DOS OR 2200 FORMAT SELECTED IF FLOPPY. CHANGED LINE 780 TO BRING YOU TO THE DISK ADDR SCREEN IF DISK SCRATCHED & RESPONDED. NO TO CONTINUE.
- 2/5/93 CFORMAT - COLLECTED PROGRAM TO SHOW ERROR IMMEDIATELY IF FORMAT FAILED.
- 2/8/93 CFORMAT - DOS FORMAT WOULD NOT COME BACK, FORMAT COMPLETED. CORRECTED LINE 813.
- 

DISK UTILITIES 1.0

- 2/16/93 CSCSICFG - CONFIGURATION SPLIT WRONG ON LINES 160 & 350, CORRECTED.
- 2/23/93 CGENPART - CORRECTED "INVALID CONFIGURATION FOR THIS CPU." w/ VLSI & 386.
- 2/29/93 CSCSICFG - CAN'T FINISH CONFIGURING IF RUN OUT OF SECTORS. ENTERING 0 NOW CAN BE USED TO END SECTOR ASSIGNMENT. ADDRESS SPLIT WRONG ON LINE 670.
- 3/10/93 CSCSICFG - IF SAVE CONFIG & RE-RUN SETUP, PASSWORD SHOWS UP IN SECTOR FIELD.
- 3/11/93 CSCSICFG - IF MORE THAN 15 ADDR. ASSIGNED, 1<sup>ST</sup> SLAVE ADDR IS LOST & ALL SUBSEQUENT ADDRESSES MOVE UP 1.
- 3/11/93 CSCSICFG - INT. X71 ON LINE 660 & 760, CONVERT S TO L# ( , ), (#####).
- 3/12/93 CFORMAT - IF DOS FORMAT CHOSEN WOULD NOT CHECK DISK FOR INDEX. DELETED LINE 705. LONG MESSAGES WOULD NOT COMPLETELY CLEAR. CHANGED LINE 700. CHANGED VERSION TO 1.07.00.
- 3/15/93 CSCTAPER - FN/TAB TO EXIT DISAPPEARS IF NON-SCSI ADDR, NOT A TAPB ADDR, OR NO TAPB FOUND FROM TAPE ADDR SCREEN.
- 3/15/93 CSCTAPEB - SAME AS ABOVE FOR CSCTAPER.
- 3/16/93 CSCTAPER - FN/TAB DISAPPEARS FROM SCREEN WHEN LOADING DIRECTORY.
- 3/25/93 CDSCFIG - IF ENTERED # > 65535, SCREEN WOULD BUMP UP A LINE & ALL FOLLOWING ENTRIES WOULD BE ON WRONG LINE.
- 3/26/93 CDSCFIG - DELETED UNNECESSARY LINE 4155.
- 4/7/93 CMOVEFIL - ADDED TEST FOR 3 BYTE ADDRESS. NOT SUPPORTED.
- 4/9/93 CTO.CREF - ADDED TEST FOR 3 BYTE ADDRESS. NOT SUPPORTED.
- 4/9/93 CINSTALL - ADDED TEST FOR 3 BYTE ADDRESS. NOT SUPPORTED.
- 4/26/93 CFORMAT - IMPLEMENTED 3 BYTE ADDRESSING.



- ~~4/1/93~~ CFORMAT - MADE CHANGES FOR VLSI TO IGNORE 3 BYTE CHANGES
- 5/21/93 CFORMAT - FIX BUG WHICH CAUSED 3 BYTE SCRATCH & CORRECTED DISPLAY ON 3 BYTE INDEX.
- 6/11/93 CFORMAT - FIX BUGS TO ALLOW FORMATTING OF PX & LVP REMOVABLES. MADE CORRECTION TO CHECK ADDRESS BIO FOR FLOPPY TYPE. MOVED MOUNT MESSAGE TO PREVENT OVERWRITE OF REMOVABLE DISK TYPE.
- 6/11/93 CTO.CREF - CHANGED VERIFY ON LINE 55 FROM (0,0) TO (0,1) TO CIRCUMVENT TURBO BUG.
- 7/22/93 CDSAPPLY - THIS FILE IS MISSING FROM THE DISK UTILITIES DISK REV 1.0, P/N 731-8015B. ADDED TO DISK UTILITIES 1.1 & TO MY MASTER 1.0.
- CDS CFG - AFTER KEYING SF'10 TO BEGIN PROCEDURE TO APPLY A NEW CONFIGURATION, AFTER RESPONDING ~~ENTERED~~ TO ENTERING A REMARK FOR HARD COPY PRINTOUT, PROGRAM WOULD NEXT PREMATURELY ASK "APPLY Y OR N". CHANGE WORD "APPLY" TO "CONTINUE".
- 8/9/93 CFORMAT - LVP DSPD COMES UP AS PX TYPE, CORRECTED.
- 9/17/93 CDS CFG - CONTINUE MESSAGE OVERWRITES TABLE, FIXED. WOULD NOT ACCEPT 'M' & 'S' FOR MASTER/SLAVE, CORRECTED.
- 1/17/94 CDS CFG - CHANGED MIN O/S REQUIREMENTS ON LINE 4315 FOR 3 BYTE ADDRESSING.
- 1/20/94 CFORMAT - MADE VER 2.00 COMPATIBLE TO 386. MUST NOW REMOVE REM% ON 935 FOR 3 BYTE SCRATCH.
- 1/20/94 CSYSTEMPB - CHANGED HEADING TO "DISK UTILITIES 1.1 .....: 1994"
- 1/20/94 C.DISK - CHANGED HEADING TO "DISK MANAGEMENT UTILITIES. (RELEASE 1.10)".
- 1/21/94 CVERCPUB/C/R - ADDED MESSAGE 'DEVICE NOT SUPPORTED' FOR NON-TURBO

### DISK UTILITIES 1.1

- 1/27/94 CDOSFORM - CORRECTED ERROR MESSAGE FOR FORMAT FAILURE ON DRIVE B
- 1/28/94 CCLOC - SPLIT LINE 470 CREATING 475 TO ALLOW CONVERSION TO 'NEW' FORMAT.
- CDAVFU - SPLIT LINE 930 CREATING LINE 935 TO ALLOW CONVERSION TO 'NEW' FORMAT.
- 1/2/94 CDS CFG - CORRECTED DS DEFAULTS TO USE MASTER & SLAVE ADDRESSES LIKE SWITCHES WOULD. CORRECT INPUT ON WARNING TO REACT ACCORDING TO KEYS. INSERTED LINE TO RECOGNIZED 32 MB MICROPOLIS.

### DISK UTILITIES 1.1

- 3/9/94 CINSTALL - UPDATED FOR CS/386 O/S 1.3 W/ 3 DISKETTES & 70 FILES. UPDATED MESSAGE TO BE MORE DESCRIPTIVE IF O/S DISK NOT INSTALLED.

- 3/10/94 CINSTALL - UPDATED FOR CS/TURBO INSTALL W/ REL 1.30.01 FROM 1.2M OR 3 360K.
- 3/2/95 CINSTALL - ON TURBO WOULD INDICATE C386 AS O/S FILENAME ON SOURCE DISK WHEN SHOULD SAY C MVP.
- 3/21/95 CPSTAT - FAILS ON NON-386 CPUs W/ ERROR S19 ON LINE 28. CORRECTED.
- 4/28/95 CGENPART - COULD NOT BOOT MVP W/ 64K MEM, NO SF'15. IF LOADED CONFIGURATION W/ > 16 PARTITIONS ON CS MACHINE. WOULD HANG IN LOOP. CORRECTED.
- 8/8/95 CDOSFORM - REMOVED \ AT END OF MESSAGE TO OPERATOR TO " AND STRIKE ENTER WHEN READY"
- 8/10/95 CDOS - ADDED ADDITIONAL SYNTAX CHECKING FOR FORMAT COMMAND TO VERIFY EITHER DRIVE A OR B SELECTED AND : IS PRESENT.
- 11/14/96 CGENPART - ADDED LINE TO ALLOW BYPASS OF "RECONFIGURATION PASSWORD" ON AUTO EXECUTION.
- 11/14/97 CCLOC - FIXED TO SHOW CORRECT YEARS FROM 1991 TO 2090
- CCLOCK - FIXED TO SHOW CORRECT YEARS FROM 1991 TO 2090
- 2/20/98 CSYS MVPB - ADDED DISPLAY CLOCK TO MENU
- 3/8/99 CBACKUP - UPDATED TO ACCEPT YEAR ~~99~~

~~BACKUP~~

3/8/99 UPDATED TO ACCEPT YEAR  $\phi\phi$ . ~~B~~

DELETED :IF U9= $\phi$  THEN 920 FROM END OF LINE 900

## @CLOC

11/28/94 WOULD NOT CONVERT TO NEW FORMAT. LINE 470 TOO LONG.

FIX: SPLIT LINE 470 CREATING LINE 475

11/14/97 ONLY DISPLAY YEARS 1900 THROUGH 1999 ON CALENDAR

FIX: 550 IF M <> INT(M) OR M < 1 OR M > 12 THEN 530: IF D <> INT(D) OR D < 1  
OR > 31 THEN 530: IF Y <> INT(Y) OR Y < 0 THEN 530: IF Y < 100 THEN Y = Y + 1900

CHANGE TO: 550 ::: IF Y > 90 THEN Y = Y + 1900: IF Y < 91 THEN Y = Y + 2000

## @CLOCK

11/14/97 ONLY DISPLAY YEARS 1900 THROUGH 1999. HANGS IN LOOP AT TRANSITION FROM 1999 TO 2000.

FIX: DELETE LINE 110 110 IF DA <> "000101" THEN 115: \$BREAK: GOTO 100

4050 IF M <> INT(M) OR M < 1 OR M > 12 THEN 4030: IF D <> INT(D) OR D < 1

OR D > 31 THEN 4030: IF ~~Y <> INT(Y) OR Y < 0~~ Y <> INT(Y) OR Y < 0 THEN 4030:

IF Y < 100 THEN Y = Y + 1900

CHANGE TO: 4050 ::: IF Y > 90 THEN Y = Y + 1900: IF Y < 91 THEN Y = Y + 2000

# NOTES

11/14/97

⤴ CLOCKM D  
 ⤴ CLOCK SYS CRTS

DEL 110 IF D\$ <> "000101" THEN 115: \$BREAK: GOTO 100

DELETE LINE 110

4050 ::: IF Y < 100 THEN Y = Y + 1900

CHANGE TO 4050 ::: IF Y > 90 THEN Y = Y + 1900: IF Y < 91 THEN Y = Y + 2000

JANUARY 1, 2000 IS A SATURDAY

145

JAN 1 2000 SAT  
 FEB 1 TUE

⤴ CLOC

550 ::: ~~Y > 90~~ THEN Y = Y + 1900: IF Y < 91 THEN Y = Y + 2000

✓ CHECKED THESE YRS TO INSURE DAY OF MONTH CORRECT

	1970	1980	1990	2000	2010	2020
1		Tu	Tu	M	Sa	F ✓
2		F	W ✓	T	Su	
3		Sa	F	W	Tu ✓	
4		Su	Sa	T	W	
5		Tu	Su	Sa	Th	
6		W	M	Su	F	
7		Th	W	M	Su	
8		F	Th	Tu ✓	M	
9		Su	F	Th	T	
0	Tu	M	Sa ✓	F	W	

CLOCK 5/9/86 REL 2.7

LIST "DATE" 20 30 50 80 160 230 240 290 370 430 450

20  $U7\$( )$  - ENTERED DATETIME MMDDYYHHMMSS

REM  $U9\$( )$  - DISPLAYED DATETIME MMDDYYHHMMSS

30 REM  $U8\$(1)$  ENTER TIME  $U8\$(2)$  ENTER DATE MM/DD/YY  $V2\$( )$  DATE FROM DATEFILE

50

80  $V3\$( ) = DATE$

160

230  $DATE = V2\$( )$  PASSWORD  $U8\$( )$

240

290

370  $V2\$( ) = DATE$

430  $V3\$( ) = DATE$

550 IF  $Y < 100$  THEN  $Y = Y + 1900$

SIMPLE FIX TO 2090  
550 IF  $Y > 90$  THEN  $Y = Y + 1900$ ;  
IF  $Y < 90$  THEN  $Y = Y + 2000$

12/31/99 - 1/1/2000

TIME HANGS ON CLOCK PROG

2/28/00 - 2/29

CLOCK SHOWS 1900

DATE/TIME ✓

2/29/00 - 3/1

"

"

12/31/00 - 1/1

"

"

CLOCK 4050 IF  $Y < 100$  THEN  $Y = Y + 1900$  ; - IF  $Y > 90$  THEN  $Y = Y + 1900$  ; IF  $Y < 91$  THEN  $Y = Y + 2000$

12/31/99 - 1/1/2000

TIME HANGS AT 00:00:00

2/28/00 - 2/29/2000

✓

2/29/2000 - 3/1/2000

✓

12/31/2000 - 1/1/2001

✓

12/31/89 - 1/1/2090

✓

12/31/90 - 1/1/2091

YEAR GOES BACK TO 1991

Interoffice Memo

Date: 05/15/97  
To: Jerry Crawford  
CC: George Reinhart  
From: Mike Bahia  
Subject: Year 2000 and 2200 Systems

Jerry,

Completed testing on the 3 different 2200 Operating Systems for the year 2000. None of the 3 Operating Systems, Basic-2 O/S 3.5 for the standard 2200, CS/386 Release 1.30.00, or Turbo Release 1.30.01 use the date or time under normal operating conditions. Both date & time are system variables which can be used by the programmer in any way they would see fit. As with many systems, the date variable is 6 characters in length in the format of YYMMDD. Any program making decisions based on the last 2 digits of the year could present a problem.

The "Initialize Date & Time" Utility program (@CLOC) which comes with all 3 operating systems was used to implement the transition testing. With each system type, a Clock Program (@CLOCK) which displays the time & date & includes the current month & following month, was run on a 2nd terminal. Various programs were executing simultaneously on other terminals including a diagnostic program to test the system instruction set, a system benchmark test, a CRT exerciser, 2200 Word Processing, and a system game, Martians. Only the 2 clock related programs showed any ill affects. On all 3 Operating Systems, the following date transitions were tested with results as shown:

Transition	Inialize Date/Time (@CLOC)	@CLOCK
12/31/99 to 01/01/00	No problem	Clock stops at 00:00:00, Program goes into endless loop
02/28/00 to 02/29/00	Year shown is 1900	Year shown is 1900
02/29/00 to 03/01/00	Year shown is 1900	Year shown is 1900
12/31/00 to 01/01/01	Years shown are 1900/1901	Years show are 1900/1901

Both programs will be fixed to correctly show the year as will the clock problem with @CLOCK stopping at 00:00:00 on transition to year 2000. The fixes should be fairly simple. If you have any questions or need further information please let me know.

05/15/97

*Confidential*

# Leap Years

## The Rule

According to the Gregorian calendar, which is the civil calendar in use today, years evenly divisible by 4 are leap years, with the exception of centennial years that are not evenly divisible by 400. Therefore, the years 1700, 1800, 1900 and 2100 are not leap years, but 1600, 2000, and 2400 are leap years.

## Background

The Gregorian calendar year is intended to be of the same length as the cycle of the seasons. However, the cycle of the seasons, technically known as the tropical year, is approximately 365.2422 days. Since a calendar year consists of an integral number of whole days, a calendar year cannot exactly match the tropical year. If the calendar year always consisted of 365 days, it would be short of the tropical year by about 0.2422 days every year. Over a century, the calendar and the seasons would depart by about 24 days, so that the beginning of spring in the northern hemisphere would shift from March 20 to April 13.

To synchronize the calendar and tropical years, leap days are periodically added to the calendar, forming leap years. If a leap day is added every fourth year, the average length of the calendar year is 365.25 days. This was the basis of the Julian calendar, introduced by Julius Caesar in 46 B.C. In this case the calendar year is longer than the tropical year by about 0.0078 days. Over a century this difference accumulates to a little over three quarters of a day. From the time of Julius Caesar to the sixteenth century A.D., the beginning of spring shifted from March 23 to March 11. When Pope Gregory XIII instituted the Gregorian calendar in 1582, the calendar was shifted to make the beginning of spring fall on March 21 and a new system of leap days was introduced. Instead of intercalating a leap day every fourth year, 97 leap days would be introduced every 400 years, according to the rule given above. Thus, the average Gregorian calendar year is 365.2425 days in length. This agrees to within a half a minute of the length of the tropical year. It will take about 3300 years before the Gregorian calendar is as much as one day out of step with the seasons.

When Pope Gregory XIII instituted the Gregorian calendar in 1582, the calendar was shifted to make the beginning of spring fall on March 21 and a new system of leap days was introduced. Instead of intercalating a leap day every fourth year, 97 leap days would be introduced every 400 years, according to the rule given above. Thus, the average Gregorian calendar year is 365.2425 days in length. This agrees to within a half a minute of the length of the tropical year. It will take about 3300 years before the Gregorian calendar is as much as one day out of step with the seasons.



# THE GREGORIAN CALENDAR AND LEAP YEARS

The current year, 1996, is referred to as a "leap year" because we have inserted a "leap day" to make the length 366 days rather than the usual 365 days. The official name of the "leap day" is an *intercalary* day (with the accent on the second syllable). Intercalary is the adjective form of the verb to intercalate, which means to insert. Once this day is inserted, or intercalated, it becomes an *embolistic* day. Still another name for this extra day every fourth year is the *bissextile* day, meaning a double sixth day.

This last name is derived from the location of the intercalary day every fourth year in the Julian calendar that was put into use by Julius Caesar in 45 B.C. He realized that a *tropical* year, the interval of time between successive beginnings of spring, is about  $365 \frac{1}{4}$  days. Since agriculture was the main occupation of most people in the world until 200 years ago, adjusting the calendar year to fit the seasons seems the most reasonable way to form a calendar. The best one can do is to make the average length of a year  $365 \frac{1}{4}$  days by having 366 days in every fourth year. Caesar's intercalary day was inserted on what is now the day before February 24, thus making a second, sixth day (*bissextile*) counting back from March 1 -- the beginning of the Roman year. (The Roman days of the month were not numbered as in our modern calendar.) In 533 A.D., when the monk Dionysius Exiguus determined the year of Christ's birth and began the numbering system for years A.D., it worked out conveniently that years evenly divisible by four were leap years.

Unfortunately,  $365 \frac{1}{4}$  days is not the exact length of a tropical year. The time between successive instants of the beginning of spring is 365 days, 5 hours, 48 minutes, and 46.02 seconds. While this makes the Julian year only 11 minutes and 13.92 seconds longer than a tropical year, it amounts to spring beginning one day earlier after about 128 years. The seasons were slowly moving backward through the year under the Julian calendar.

In 1582, Pope Gregory XIII was informed by his astronomer, Christopher Clavius, that the first day of spring had fallen on March 10 of that year. Since the first day of spring had fallen on March 21 in 325 A.D. when the Council of Nicaea had established the dates for the holidays in the Christian calendar, Gregory felt it was important to put the seasons back in the same places in the calendar. He therefore declared that the next day after October 4, 1582, was October 15, 1582. He also adopted a revision in the calendar which resulted in the average length of a year being closer to the length of a tropical year. This calendar, which we now use, is called the Gregorian calendar. It differs from the Julian calendar by letting century years, such as 1600, 1700, 1800, etc., be leap years *only* if they are evenly divisible by 400. Thus, 1600 was a leap year, and 2000 will be, too, but 1700, 1800, and 1900 were not leap years.

The average length of our Gregorian year is 365 days, 5 hours, 49 minutes, and 12 seconds, still about 26 seconds longer than a tropical year. However, more than 3000 years must now pass for the seasons to move back by one day. A suggested modification to the Gregorian calendar is to eliminate as leap years those years which are evenly divisible by 4000. This would result in the length of a year, averaged over 4000 years, being only 4 seconds longer than a tropical year. Fortunately, we still have a couple thousand years before we need to give serious thought to this modification.

CDAYFU

1/28/94 WOULD NOT CONVERT TO NEW FORMAT. LINE 930 TOO LONG.

FIX: SPLIT LINE 930 CREATING LINE 935.

↳ DOS COPY

FAILS W/ D86 ON LINE 180 OF

↳ DOS COPY IF TRY TO COPY A  
FILE FROM A FLOPPY TO A DOS DISK.

COPY A:<sup>D:\</sup>CMXED B:<sup>D:\</sup>CMXED

WORKS IF COPY FROM 340

COPY C:<sup>340</sup>CMXED B:<sup>D:\</sup>CMXED

US65 ↳ DOS CP CS

# IMPORTANT MESSAGE

TO \_\_\_\_\_

DATE \_\_\_\_\_ TIME \_\_\_\_\_ A.M.  
P.M.

## WHILE YOU WERE OUT

M \_\_\_\_\_

OF \_\_\_\_\_

Area Code  
& Exchange \_\_\_\_\_

TELEPHONED		PLEASE CALL	
CALLED TO SEE YOU		WILL CALL AGAIN	
WANTS TO SEE YOU		URGENT	
	RETURNED YOUR CALL		

Message \_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

\_\_\_\_\_

Operator \_\_\_\_\_



CDOS

8/10/95

IF TYPE IN WRONG SYNTAX OR ILLEGAL DRIVE PARAMETER WITH  
FORMAT COMMAND, DOES NOT GIVE AN ERROR & DEFAULTS TO THE  
A DRIVE.

FIX: ADDED ADDITIONAL SYNTAX CHECKING FOR FORMAT COMMAND ON LINE  
460 WITH ADDITIONAL ERROR MESSAGES.

```
460 IF STR(B#,8,2)="" THEN 896: IF STR(B#,8,1)<>"A" AND  
STR(B#,8,1)<>"B" THEN 895: IF STR(B#,8,2)<>"A:" AND STR(B#,8,2)<>  
"B:" THEN 897: LOAD T "CDOSFORM"
```

```
895 PRINT HEX(ØD ØE Ø7); "INVALID DRIVE SPECIFICATION (ONLY DRIVE A OR B  
SUPPORTED WITH FORMAT)"; HEX(ØF): GOTO 180
```

```
896 PRINT HEX(ØD ØE Ø7); "REQUIRED PARAMETER MISSING (DRIVE  
DESIGNATION & COLON)"; HEX(ØF): GOTO 180
```

```
897 PRINT HEX(ØD ØE Ø7); "PARAMETER FORMAT NOT CORRECT (DRIVE  
DESIGNATION & COLON)"; HEX(ØF): GOTO 180
```

# CDOSFORM

1/27/94 If FAILS FORMATTING DRIVE B COULD RETURN MESSAGE "GENERAL FAILURE  
ERROR READING DRIVE A".

FIX: CHANGED <sup>ERROR MESSAGE ON</sup> LINE 480 AS FOLLOWS:

480 % GENERAL READ ERROR OR BAD WRITE

8/8/95 If TRY TO FORMAT ~~E~~, COMES BACK WITH MESSAGE TO "INSERT NEW  
DISKETTS FOR DRIVE A: AND STRIKE ENTER WHEN READY"

~~WOULD ALWAYS DEFAULT TO DRIVE A OR B IF DRIVE LETTER INCORRECT.~~

~~ADDED SYNTAX CHECKING FOR FORMAT COMMAND & REMOVED \ FROM~~

~~REMOVED \ AT END OF MESSAGE.~~

470 % and strike ENTER WHEN READY

Package Subject: DOS Utilities

Item Title: DOS Utilities

Mike

I will check the error when using drive b. There is a hidden key to bring the last command back. This is done by pressing Cancel/Edit key then SF'3 on 2200 keyboard or sf'4 on a pC emulation which is SF'3 in reality.

Regards John Baxi

----- Original Memo -----

To: Kirit Baxi  
Subject: DOS Utilities

From: Mike Bahia  
Date Sent: 10/08/92

John,

During the last few days I have been doing some testing with the DOS Utilities. While it was fresh on my mind I had a couple of suggestions.

1. In testing FORMAT, I was using drive B. Was formatting a 360K disk in a 1.2 Meg which I knew was illegal. However, it did go out to format the disk but when it failed it indicated a failure with drive A. The message was, "General Failure error reading drive A" "Abort, Retry, Fail?"

FIXED 11/27/94

I believe this should have indicated the error was reading drive B. Drive A is address D30 and has the DOS Utilities as found on the latest Turbo O/S, 1.18. Drive B is D20. Also, what does Fail mean in the 'Abort, Retry, Fail?' line. It seems to be the same as Abort. Doesn't seem to be necessary.

2. Sometimes when you do get an error you need to rerun the Utility to get the prompt back. It would be nice to have a SF' key to bring the prompt back. What do you think?

PS: Initially had a lot of problems because I did not realize a space was not needed between the drive designation and the filename for those commands that use both. This was causing most of my problems. This is where a SF key to bring back the DOS prompt would have been handy.

Best regards,  
Mike

# LINE/COMMAND TRAP VER FOR MARINE BIOLOGICAL LABS

DSTAPEB 1/3/92 REL 2.6

A# = COMMAND L# = LINE#

1430 TAPE COMMAND ERROR  
LIST T "1430" 810, 840, 850, 870, 880, 920, 940, 960, 1000, 1020, 1030, 1130, 1166,  
1200, 1240, 1250, 1330

810 REM% REWIND TAPE: GOSUB '201 ("REWINDING TAPE"): \$G10REWINDTAPE#1: A# =  
"REWIND": L = 810: IF STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430

840 REM% ERASE TAPE: GOSUB '201 ("ERASING TAPE"): \$G10ERASETAPE#1: A# =  
"ERASE": L = 840: IF STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430

850 REM% POSITION TO BEGINNING OF TAPE DIRECTORY: GOSUB '201 ("POSITIONING TO  
TAPE DIRECTORY"): STR(G#, 3, 3) = HEX(00 00 00): \$G10SEEKDIRECTORYBLOCK#1:  
A# = "SEEK DIR": L = 850: IF STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430

870 REM WRITE BLOCK TO TAPE BUFFER: GOSUB '203: A# = "WR DIR LABEL": L = 870: IF  
STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430

880 REM WRITE FILE MARK: \$G10WRITEFILEMARK#1: A# = "WR DIR FM": L = 880: IF STR  
(G#, 6, 3) <> HEX(00 00 00) THEN 1430

920 REM% RETENSION TAPE: GOSUB '50 (HEX(0E), "RETENSIONING TAPE"): \$G10RETENSION  
TAPE#1: A# = "RETENSION": L = 920: IF STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430: TA = "T"

940<sup>REM</sup> POSITION TO BEGINNING OF TAPE DIRECTORY: GOSUB '201 ("POSITIONING TO TAPE  
DIRECTORY"): STR(G#, 3, 3) = HEX(00 00 00): \$G10SEEKDIRECTORYBLOCK#1: A# = "SEEK  
DIR": L = 940: IF STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430

960 REM READ TAPE DIRECTORY LABEL (1<sup>ST</sup> BLOCK): GOSUB '202: A# = "READ DIR LABEL":  
L = 960: IF STR(G#, 6, 3) <> HEX(00 00 00) THEN 1430

1000 REM CHECK FOR END OF DATA: A# = "CHK END/  
of DATA": L = 1000: IF STR(G#, 6, 3) <>  
HEX(17 00 00) THEN 1430

1020 M# = "POSITIONING TO LAST BLOCK": GOSUB '201 (M#): STR(G#, 3, 3) = C#: IF C# > HEX(00  
00 00) THEN STR(G#, 3, 3) = SUBHEX(00 00 01): IF C# < HEX(00 3C 00) THEN STR(G#, 3, 3) =

HEX(00 00 00): \$G10SEEKBLOCK#1: A# = "SEEK LAST BLK": L = 1020: IF STR(G#, 6, 3) <> HEX(00 00 00)  
THEN 1430



THESE COMMANDS ARE FOR THE LOGIC BIOLGICALS CASE

```

1030 M# = "POSITION TO END OF DATA": GOSUB '201(M#): $G10SEEKENDOFDATA#1:
: A# = "SEEK END/DATA": L = 1030: IF STR(G#,6,3) <> HEX(00 00 00) THEN 1430
1130 $G10BACKUPSECTORS#1: A# = "BD SECTORS": L = 1130: IF STR(G#,6,3) <> HEX(00 00 00) THEN 1430
1166 REM WRITE BLOCK TO TAPE BUFFER: GOSUB '203: A# = "WR BLK1": L = 1166: IF STR(G#,6,3) <>
HEX(00 00 00) THEN 1430: NEXT Z
1200 REM WRITE BLOCK TO TAPE BUFFER: GOSUB '203: A# = "WR BLK2": L = 1200: IF STR(G#,6,3) <>
HEX(00 00 00) THEN 1430
1240 REM WRITE BLOCK TO TAPE BUFFER: GOSUB '203: A# = "WR BLK3": L = 1240: IF STR(G#,6,3) <>
HEX(00 00 00) THEN 1430
1250 REM % WRITE FILE MARK: $G10WRITEFILEMARK#1: A# = "WR DATA FM": L = 1250: IF STR(G#,6,3)
<> HEX(00 00 00) THEN 1430
1330 REM % REWIND TAPE: GOSUB '201("REWINDING TAPE"): $G10REWINDTAPE#1: A# = "REWIND 2":
L = 1330: IF STR(G#,6,3) <> HEX(00 00 00) THEN 1430

```

```

1430: IF STR(G#,7,2) = HEX(00 00) THEN 1440: M# = "TAPE COMMAND ERROR": GOTO 1700
1700 REM % NONRECOVERABLE ERROR: M1# = ". TAPE BACKUP ABORTED": IF D3# <> " " THEN M1# =
M1# & " COPYING / " & D3# : GOSUB '50(M1#, M#): PRINT HEX(07);
PRINT AT(20,0); "LAST TAPE FUNCTION: "; A#; " LINE "; L; " ERROR CODE = "; STR(G#,6,1);
" COMMAND ERROR = "; STR(G#,7,2);
1705 PRINT AT(21,0); "STR(G#,6,3) = "; HEXPRINT STR(G#,6,3): PRINT AT(21,25); " ERROR
CODE IN HEX = "; HEXPRINT STR(G#,6,1): PRINT AT(21,50); " COMMAND ERROR IN HEX = ";
HEXPRINT STR(G#,7,2)

```

## @DSCFIG

11/15/92 CHANGED ALL REFERENCES TO 'SURFACE' IN ALL SCREENS TO 'ADDRESS'.

LINES 90, 350, 1145, 4030, 4290, & 8515

11/23/92 REMOVED PERIODS FROM DS AND ADDED ? TO QUESTIONS.

LINES 4030, 4132, 4160, 4215, 8515

12/2/92 IF ENTER 'N' WHEN ASKING IF YOU WANT TO APPLY Y OR N? \_

TO APPLYING CONFIGURATION PROGRAM RETURNS :\_. SAME RESPONSE FROM ANY KEY BUT 'Y'.

CHANGED LINE 8080 AS FOLLOWS:

```
WAS 8080 K$=" ": PRINT AT (21,0,80);: INPUT "Apply Y or N"-K$: IF K$ <> "Y" THEN RETURN  
TO 8080 K$=" ": : INPUT "Apply Y or N?"-K$: IF K$="Y" THEN 8150:  
IF K$ <> "N" THEN 8080: GOTO 40
```

ADDED WARNING WHEN APPLYING CHANGES. ADDED LINE 8150 & 8155.

12/3/92 ADDED WARNING TO BACKUP WHEN KEY SF'10 TO APPLY CONFIG.

SPLIT LINE 8010, MAKING 8012. PUT MESSAGE ON 8010.

CHANGED MESSAGE FOR APPLY PROCEDURE COMPLETED. SEE LINE 9010.

12/5/92 MOVED PASSWORD TO PRINT AT LINE 2 INSTEAD OF 1. SEE LINE 8020.

MOVED "CONFIG FILENAME" TO PRINT AT LINE 2 INSTEAD OF 1. SEE LINE 8030.

ADDED SPACE AFTER "ASSIGNMENTS" TO REMOVE THE WORD EXIT.

— USING DEFAULT CONFIG GAVE ILLEGAL MESSAGE. CHANGE IF/THEN

ADDRESS FROM 4384 TO 4390 ON LINE 4132 TO MATCH DS UTIL 2. OF WHICH WORKS.



1/15/93 ADDED NOTE TO INDICATE IF >65535 SECTORS ASSIGNED TO DISK MIN O/S  
386 1.2 OR TURBO 1.1 REQUIRED. CREATED NEW LINE 4315

4315 IF X < 65536 THEN 4320: PRINT HEX(010E); AT (23,0); "MIN O/S 386 1.2  
OR TURBO 1.1 REQUIRED w/ >65535."

ADDED PRINT AT (23,0,55)

AT BEGINNING OF LINE 4290 TO CLEAR O/S LEVEL MESSAGE.

ADDED CODE TO PREVENT OVERWRITE OF OLD CONFIG FILE WITHOUT NOTICE.

NEW LINES 4452 IF E <> 2 THEN 4440: KA = " ": PRINT AT (23,0,55); "DATA  
FILE "; FB; " ALREADY EXISTS. "; INPUT "OVERWRITE, Y/N?" - KA

4453 IF KA = "N" OR KA = "n" THEN 4440: IF KA = "Y" OR KA = "y" THEN 4455:  
GOTO 4452

DUE TO PRINT AT PROBLEM WITH TURBO WHICH CONFUSES SCREEN IF  
SECTOR VALUE EXCEEDS 65535 ADDED TURBO TEST EXIT ON LINE 4315  
AFTER FIRST :

JA = BPSTAT(1): IF STR(JA,9,1) = "T" THEN 4320:

3/25/93 IF ENTERED A SECTOR VALUE > 65535 ON A 386 OR VLSI, SCREEN WOULD  
BUMP UP 1 LINE & ALL FUTURE SECTOR ENTRIES WOULD NOT LINE UP WITH PROPER ADDRESS.

ALSO VALUE > 65535 NOT ACCEPTED ON VLSI OR 386, ~~WAS~~ NOW. MUST REENTER.

4315 IF X : JA: IF STR: PRINT HEX(010E); AT (23,0); "MIN O/S 386 1.2/TURBO 1.1  
FOR >65535. " ;: GOTO 4295

3/26/93 DELETED LINE 4155 WHICH PUT UNNEEDED MESSAGE "RUN - ACCEPT SCREEN?" ON  
CRT WHEN ASKING FOR MASTER OR SLAVE ADDRESS.

7/27/93 CHANGE LINE 8080 SUBSTITUTING CONTINUE FOR THE WORD APPLY IN THE INPUT  
STATEMENT. APPLY WAS BEING PREMATURELY USED. AFTER MESSAGE FOR HARD COPY.

9/17/93 IN PROCESS OF APPLY PROCEDURE, AFTER RESPONDING TO 'HARD COPY TO PRINTER' AND REMARK, THE SCREEN UPDATES WITH 'CONTINUE Y OR N?' BUT OVERWRITES THE LAST LINE OF ADDRESSES, D x E.

CHANGE LINE:

```
8080 K# = " " : PRINT AT (23, 0, 80); : LINPUT .....
```

9/17/93 MASTER/SLAVE WONT ACCEPT SMALL S/M.

CHANGE LINE:

```
4165 AB = " " : IF K# = "M" OR K# = "m" THEN AB = "MASTER" : IF K# = "S" OR K# = "s" THEN AB = "SLAVE" : IF AB = " " THEN 4160 : PRINT AT (20, 13); AB;
ADD 4167 IF AB = "MASTER" THEN K# = "M" : IF AB = "SLAVE" THEN K# = "S"
```

1/17/94 CHANGED LINE 4315 TO INDICATE MINIMUM O/S REL FOR 3 BYTE ADDRESSING IS TURBO 1.30.01 OR 1.25.

```
4315 IF X < 65536 THEN 4320 : JA = #PSTAT(1) : IF STR(J#, 9, 1) = "T" THEN 4320 : PRINT HEX(01 0E); AT (23, 0); "MIN O/S TURBO 1.30.01 OR 1.25 FOR > 65535. " ; : GOTO 4295
```

2/2/94 IF USING A 140M DRIVE W/ 'USE DS DEFAULTS', W/ A 2<sup>ND</sup> DRIVE <sup>BOTH</sup> WOULD BE CONFIGURED ON THE MASTER SIDE & 4 ADDRESSES WOULD BE LOST. SET UP DEFAULTS SO DRIVES 3 & 4 ALWAYS WILL COME UP AS SLAVE ADDRESSES, 1 & 2 AS MASTER ADDRESS, SAME AS OLDER FROM USING SWITCHES. ALSO DID NOT VERIFY EXISTENCE OF DRIVE SELECT 1.

```
Fix: 4390 REM: : : FOR I = 1 TO 4 : IF WZ#(I) = " " THEN 4398 : IF WZ#(I) = " " THEN 4394 : IF I = 1 THEN STR(WZ#(I), 6) = HEX(01) : IF I = 2 THEN DO : STR(WZ#(I), 6) = STR(WZ#(1), 4) ADD HEX(01) : ENDDO
```

```
4391 IF I = 3 THEN STR(WZ#(I), 6) = STR(WZ#(1), 4) HEX(41) : IF I = 4 THEN DO : STR(WZ#(I), 6) = HEX(41) : IF STR(WZ#(3), 6) = HEX(41) THEN STR(WZ#(I), 6) = STR(WZ#(3), 4) ADD HEX(41) : ENDDO : K# = STR(WZ#(I), 6) : IF K# > " " THEN K# = K# ADD HEX(C6) : X = MAX(VAL(K#), 1) : REM OVER
```

CONT:

4394 NEXT I: GOTO 4400

4398 STOP "ILLEGAL CONFIGURATION!! MUST HAVE A DRIVE 1 TO CONFIGURE.

UNCHESTEL DRIVES NEED TO BE PROPERLY CONFIGURED. CORRECT CONFIGURATION BEFORE PROCEEDING."

AT WARNING WHEN STEPPING THROUGH APPLY ROUTING, ANYTHING BUT A CAPITAL 'Y' RESPONSE TO CONTINUE RETURNS YOU TO BASE ADDRESS SCREEN.

FIX 8150 DIM I#1: PRINT "...: INPUT ... CONTINUE, Y OR N?" - I# : IF I# = "Y" OR I# = "y" THEN 8160

8155 IF I# = "N" OR I# = "n" THEN 40: GOTO 8150

32M MICROPOLIS MAY NOT BE RECOGNIZED AS A CORRECT SW SETTING FOR SOME PROM VERSIONS. PUT IN FOLLOWING LINE WITH A REM AS DID NOT HAVE A 32M MICROPOLIS TO TEST.

1116 REM IF P=9 THEN P=6: REST OF LINE UNCHANGED.



# CDS CFG BUGS

3/25/93 IF USE DS DEFAULTS <sup>OR CREATE OWN CONFIG</sup> TWICE IN A ROW USING THE SAME CONFIGURATION FILE NAME, FAILS 2<sup>ND</sup> TIME W/ MESSAGE "ILLEGAL SECTOR ADDRESS OR NO PLATTER". HAPPENS ON TURBO, 386, & VLSI.

4/9/93 TESTED ON 386 W/ 32 PRAM + NO PROB  
TESTED ON TURBO W/ 32 PRAM + OK  
TESTED ON TURBO W/ 48 PRAM + OK

CIRCUMVENTION: OPEN & SHUT FLOPPY DOOR OR RESET AND REBUN. 4/8 CANNOT DUB.  
2/2/94 CANNOT DUB.

3/25/93 IF USE EXISTING CONFIGURATION FILE & RESPOND NO TO OVERWRITE SCREEN BUMPS UP A LINE. (TURBO, 386, & VLSI) LINE 4452

3/26/93 MASTER/SLAVE WON'T ACCEPT SMALL S/M. FIXED LINE 4165 OR IF KB="m"  
GETTING PS6 SUBSCRIPT OUT OF RANGE ON LINE 4200. IF WH(IO) OR IF KB="s"

3/25/93 WITH CONFIGURED 140M ON DRIVE SELECT 1 & UNCONFIGURED 64M ON DRIVE SELECT 3, WHEN USED DS DEFAULTS ASSIGNED ADDRESSES D11-D14 TO THE 64M (DRIVE 3) & ADDRESSES D15-D18 TO THE 140M.  
IF MAKE 64M DRIVE 1 & 140M DRIVE 3, DRIVE 3 IS ASSIGNED ADDRESSES D11-D18 & THE 64M IS LOST (USING DS DEFAULTS). FIX LINE 4390, 4391 2/2/94

9/16/93 WHEN PREPARING TO 'APPLY' A CREATED CONFIGURATION AFTER RESPONDING TO 'HARD COPY TO PRINTER' & 'REMARK FOR HARD COPY' 'CONTINUE Y OR N?' WIPES OUT LAST LINE OF ADDRESSES. (D x E LINE)

Fix: CHANGE LINE 8080

9/16/93 W/ 64M AS DRIVE 1 & 140M AS DRIVE 3, CREATED 4 64MEG SURFACES FOR 64M & SEVERAL SURFACES FOR 140, ALL MASTER ADDRESSES FROM D25-D28.  
ONLY APPLIED ADDRESS TO DRIVE 3. WHEN LOOKED AT CONFIG 64M APPEARED AS UNCONFIGURED & 140M AS ADDRESS D21-D29 INSTEAD OF D25-D28.

1/25/94 NPF IF CHANGE ADDRESSES OF EXISTING DRIVES GOOD IDEA TO POWER OFF FIRST TO CLEAR CABLES WHICH CAN CAUSE WEIRD PROBLEMS. FIXED 2/2/94



CASEDEFT - BOTH 64 DRIVE 1 & 140 DRIVE 2 DEFAULTS ✓

~~Z~~ ~~RAVILI~~

LOOKS CORRECT

D21	16M	D61	10M	
D22	16M	↓	D6E	10M
D23	16M			
D24	16M			

~~R~~

APPLIED BOTH → SEEMS OK

APPLIED DRIVE 3 ONLY SEEMS OK

APPLIED DRIVE 1 ONLY SEEMS OK

CASE 2 64M D21-D24 ALL 16M 140M D25-D28 16M D2D 6M

LOOKS CORRECT

APPLIED ONLY DRIVE 3

D61 I91 SEEMS CORRECT

140M NOW DRIVE 1 64M NOW DRIVE 3

CONFIG SHOWS 140M D21-D28 16M D29 6M 64M D2A-D2D 16M

LOOKS OK w/ CDSCFIG LISTS SEEM CORRECT

DIS DEFAULTS	140M	64		
	D21	10M	D2B	16M
	↓	↓	↓	
	D2A	10M	D2E	16M

64M SHOULD BE D61-D64

140 SHOULD BE D21-D25

CONFIG SCREENS MATCH LIST MATCHES CONFIG



USE DS DEFAULTS? Y HALFSTEP 140M - DRIVE 1 64M - DRIVE 3

4132 IF K#="Y" THEN 4390

4390 REM SET W#(32)3 SIZES

INIT(00)W#( ): MAT W# = ZER: FOR I = 1 TO 4

FOR I = 1 TO 4 4 DRIVES

IF WZ#(I) = " " : 4394

CHECK ENTRY IN WZ#(I) TO SEE IF DRIVE EXISTS

WZ# = [00 98 00 DE 37 01] 20 20  
 [20 20 20 20 00 FE 00 04] 20 20  
 [20 20 20 20 20 20 20 20]

K# = STR(WZ#(I), 6)

01 SET K# TO 6TH BYTE OF WZ#(I)

IF K# > " " THEN K# = K# + HEX(C6)

01 IF K# IS BLANK ADD HEX(C6)

X = MAX(VAL(K#), 1)

1 RBM X = START DRIVE

X IS FIRST ADDRESS

4392 FOR Z = 1 TO VAL(STR(WZ#(I), 4))

14 ADDRESSES

Z IS # OF ADDRESS FOR DRIVE

W#(X+Z-1) = STR(WZ#(I), 3)

W#(2) = 00 98 00

W#(1) = 00 98 00

ASSIGN SECTORS TO W# (FOR EA ADDR) ADDR FOR EA DRIVE

W# = 14 DIS

01 01

W# = 01 00 1...

IF STR(DI#(I), 1, 1) <> "Z" THEN STR(W5#(X+Z-1, 1)) = BIN(I)

DI#(2) = 37 82 0A 01 01 0A

36 82 04 03 0B 04

31 96 00 02 00 00

31 96 00 04 00 00

NEXT Z

W#(I) = VAL(STR(WZ#(I), 5)) - 47

37 HEX = 55

W#(1) = 8

? W#(I) = 5TH BYTE OF WZ#(I) - 47

NEXT I

4390 IF WZ#(I) = " " : 4394

WZ#(2) = 20 20 20 20 20 20

4394 NEXT I

IF WZ#(I) = " " : 4394

WZ#(3) = 00 FE 00 04 36 0B

CHECK ENTRY IN WZ#(I) TO SEE IF DRIVE EXISTS

K# = STR(WZ#(I), 6)

0B SET K# TO 6TH BYTE OF WZ#(I)

WHICH IS 0B (1ST ADDR.)

IF K# > " " THEN K# = K# + HEX(C6)

X = MAX(VAL(K#), 1)

04 X IS FIRST ADDR

4392 FOR Z = 1 TO VAL(STR(WZ#(I), 4))

4 ADDRESSES

W#(X+Z-1) = STR(WZ#(I), 3)

W#(2) = 00 FE 00

W#(1) = 00 FE 00

ASSIGN SECTORS TO W# (FOR EA ADDR) ADDR ENTRY FOR EA DRIVE

W# = 03

IF STR(DI#(I), 1, 1) <> "Z" THEN STR(W5#(X+Z-1, 1)) = BIN(I)

W5# = 01 01 01 01 01 01 01 01 01 01 03 03 03 03

NEXT Z

W#(I) = VAL(STR(WZ#(I), 5)) - 47

HEX 36 = 54

W#(3) = 7

NEXT I

4390 IF WZ#(I) = " " THEN 4394

4394 NEXT I

4400 CREATE DRIVE CONFIG: BYTES 001-032 CONSTANT: 011-013 = "TBO": 14 # of PLATTERS: 15 START ADDR

4410 BYTES 033-256 1 K BYTE ENTRY | DRIVE SSS + NNN: WHERE # (N), 3, 3) = NNN = # OF SECTORS

4420 W5 # 32 BY SURFACE DRIVE 1, 2, 3, 4: W#(32) # 3 BY SURFACE # OF SECTORS

4430 PRINT

4440 FA = "CDEFAULT": LINPUT "FILGNAM6" FA CASE 6

4450 LIMITS T#0, FA, A, B, C, E: E = ERR

4452 IF E <> 2 THEN 4455

4455 IF E = 1 : 4440

IF E = 0 : DATASAVE DC OPEN T#0, (6) FA <sup>340</sup> <sup>5 SECTORS</sup> <sup>CASE 6</sup> OPEN DATAFILE CASE 6

4460 LIMITS T#0, FA, A, B, C, E <sup>1109 1109 1 2</sup>

DATALOADDC OPEN T#0, FA

WI = A

4470 FOR W = 1 TO 4

KA = BIN W <sup>KA = 01</sup>

INIT(00) R#( ) = X = 0

FOR I = 1 TO 31

IF STR(W5 # I, 1) = KA THEN DO

X = X + 1

STR(R#(X \* 2), 4, 3) = W#(I) : ENDDO <sup>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31</sup>

WRITE SECTOR SIZE FOR EA ADDR INTO R#

01 01 01 01 | 01 01 01 01 | 01 01 03 03 | 03 03

NEXT I

$WZ\#$       $25$       $190$       $326$       $1170$       $4387$       $4390$       $4392$

\* NO GET  
 DRIVE NEXT  
 DRIVE  
 ASSIGN  
 SECTION  
 TO WB(FOR EA ADDR)

25 DIM WZ\$(4)6

190 WZ\$ = " "

326 WZ\$(WZ) = BIN(WI,3) & BIN(P9) & GI# & K#

1170 WZ\$(I) = BIN(Z,3) & BIN(P(P)) & STR(DI\$(I),,1) & STR(DI\$(I),5,1)

4387 STR(WZ\$(I),6,1) = K#

4390 IF WZ\$(I) = " " THEN 4394

4392 W\$(X+Z-1) = STR(WZ\$(I),,3)



USE DS DEFAULTS? Y HALT(STEP 64M = DRIVE 1 140M = DRIVE 3

4132 IF KB = "Y" THEN 4390 ✓

4340 REM SET WB(32)3 w/ sizes

INIT (00)WB(), WSA: MAT W0=ZER

W2#() = 00 FE 00 04 36 01  
 20 20 20 20 20 20  
 00 98 00 0E 37 05  
 20 20 20 20 20 20

FOR I = 1 TO 4

IF WZ#(I) = " " THEN 4394 NO DRIVE

K# = STR(WZ#(I), 6) START ADDR 01 FOR DRIVE 1

IF K# > " " THEN K# = K# + HEX(C6)

X = MAX(VAL(K#), 1): X = 1 FIRST ADDR

4392 FOR Z = 1 TO VAL(STR(WZ#(I), 4)) # OF ADDRS FOR CURRENT DRIVE

4 times

WB(X+Z-1) = STR(WZ#(I), 3)

WB(1) = 65024  
 0101

IF WINL THEN EXCL ENTER DRIVE

DIA() = 36 82 04 01 01 04  
 37 82 0A 03 05 0A  
 31 96 00 02 00 00  
 31 96 00 04 00 00

IF STR(DIA(I), 1, 1) <> "Z" THEN STR(WSA, X+Z-1, 1) = BIN(I)

NEXT Z

W#(I) = VAL(STR(WZ#(I), 5)) - 47 W#(1) = 7

NEXT I

4390 I=2 IF WZ#(I) = " " THEN 4394 NO DRIVE 2

4394 NEXT

4390 I=3 IF WZ#(I) = " " THEN 4394

K# = STR(WZ#(I), 6) START ADDR 05 FOR DRIVE 3

IF K# > " " THEN K# = K# + HEX(C6) K# = 05

X = MAX(VAL(K#), 1) X = 5

4392 FOR Z = 1 TO VAL(STR(WZ#(I), 4))

WB(X+Z-1) = STR(WZ#(I), 3) W#5 = 38912

IF STR(DIA(I), 1, 1) <> "Z" THEN STR(WSA, X+Z-1, 1) = BIN I

ADDED # 1116 IF P=9 THEN P=6

SPLIT 4390

4391 IF :

A WARNING CONTINUE Y OR N ANYTHING NOT Y TAKE YOU TO BASE ADDR

8150

8155

SWAPPED 140 TO DRIVE 1 (HAD ADDR D61-D6E & 64M TO DRIVE 3 (HAD ADDR D21-D24)

HAD CABLES WRONG ONCE & SW SETTINGS

RAN PROG

KEY SF 15 TO START SETUP

USE DEFAULTS Y

FILE NAME CASE 8 RET

8432 : : : W\$(J) = STR(R\$(Z+I), 4)

J = 33

Proposed DS Address Assignments  
2 Winchester with sectors available



Master disk Address		Catalog Maximum	Slave disk Address		Catalog Maximum
D21	1	65024	D61	3	38912
D22	1	65024	D62	3	38912
D23	1	65024	D63	3	38912
D24	1	65024	D64	3	38912
D25	0	0	D65	3	38912
D26	0	0	D66	3	38912
D27	0	0	D67	3	38912
D28	0	0	D68	3	38912
D29	0	0	D69	3	38912
D2A	0	0	D6A	3	38912
D2B	0	0	D6B	3	38912
D2C	0	0	D6C	3	38912
D2D	0	0	D6D	3	38912
D2E	0	0	D6E	3	38912

Configuration created in file CASEDEFT  
Key Reset and sf '10 to apply

FN/TAB - Exit

D21-4 64m ✓

Proposed DS Address Assignments  
2 Winchester with sectors available

Data as read from file                      write this for drive f

01000000000000000000000054424F040107  
2020200000000000000000000000000000  
00000000FE0000000000000000000000  
00000000FE0000000000000000000000  
00000000FE0000000000000000000000  
00000000FE0000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000  
00000000000000000000000000000000

- 1 65014
- 2
- 3
- 4

Apply Y or N.





CASE2 1/28/94

Proposed D S Address Assignments  
2 Winchester with sectors available

Master disk Address		Catalog Maximum	Slave disk Address		Catalog Maximum
D21	1	65024	D61	0	0
D22	1	65024	D62	0	0
D23	1	65024	D63	0	0
D24	1	65024	D64	0	0
D25	3	65024	D65	0	0
D26	3	65024	D66	0	0
D27	3	65024	D67	0	0
D28	3	65024	D68	0	0
D29	3	65024	D69	0	0
D2A	3	65024	D6A	0	0
D2B	3	65024	D6B	0	0
D2C	3	65024	D6C	0	0
D2D	3	24576	D6D	0	0
D2E	0	0	D6E	0	0

D21-4  
64M

Master disk            Catalog  
Address                Maximum

Slave disk            Catalog  
Address                Maximum

Data as read from file

write this for drive 1

01000000000000000000000054424F040107  
20202000000000000000000000000000  
00000000FE00000000000000000000000  
00000000FE00000000000000000000000  
00000000FE00000000000000000000000  
00000000FE00000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000  
000000000000000000000000000000000

SAME AS CASE DEFT

Apply Y or N .



STOP 190 DRIVE 1 = 64M DRIVE 3 = 140M

(HEX(06),  
GOSUB '50, 'LEGAL')

RUN  
BASE ADDR D20 RETURN

GETS DEVICE STATUS

STOP 190 WZ# = ALL(20)

INIT(00)W#()

WZ#() = " " ALL 20:

MAT W = ZER : W2, W3 = 0

200 IF R# > "01" THEN V1 = 1 R# = 40 PROM VER 1.0

IF V# = 1 THEN GOSUB 3010 V = 0

21 IF R# <= "32" THEN 250 PROM VER > 1.0 ≤ 3.2

GOSUB 1020 GET DRIVE SW STATUS

1020 G# = STR(P#, 1, 1) OR HEX(20) → G# = 30 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20

1030 E#, STR(G#, 2, 7) = ALL(00) → P# = 10 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E  
10 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E

#G10READDRIVESTATUS#2(, G#)G#; STR(E#, VAL(STR(G#, 5, 1)))

1040 PRINT

1050 D#( ) = E#

G# = 30 00 00 00 12 00 00 00 00 12 20 20 20 14 00 20 20

~~E# = SAME AS ABOVE~~

FOR I = 2 TO 5

D#(I-1) = D#(I) & BIN(I-1) & HEX(0000)

E# = 31 82 01 36 82 04 31 96 00 37 82 0A 31 96 00 45

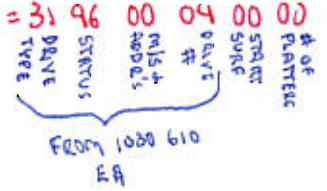
82 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00

D#(1) = 36 82 04 01 00 00

D#(2) = 31 96 00 02 00 00 NO DRIVE

D#(3) = 37 82 0A 03 00 00

D#(4) = 31 96 00 04 00 00 NO DRIVE



1060 REM MOVE WINC INTO D#( )

1070 REM

1080 REM

1085 N1, N2 = 0

FOR I = 1 TO 4

D#(I) = D#(I) 1. 36 82 04  
2. 31 96 00

X, Z = VAL(STR(D#(I), 3)) 1. 4 2. 0 3. 10

IF Z = 0 THEN 1100

IF Z > 64 THEN Z = Z - 64

IF STR(D#(I), 1) < "2" THEN 1090 36 = "6"

IF SLAVE ADDR SUBTRACT HEX(40)

IS IT A WINC DRIVE (3-10)?

1090 IF X > 15 THEN 1095 X = 4

FOR SLAVE ADDR (> 15)?

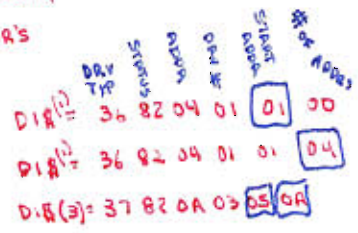


X = NI + 1 1. x = 1 3. x = 5 START ADDR

NI = NI + Z 1. NI = 2 2. NI = 14 # OF ADDR'S

GOTO 1100

```
1100 BIN(STR(DI$(I), 5)) = X
BIN(STR(DI$(I), 6)) = Z
NEXT I
```



START SURFACE ADDR

```
1110 FOR I = 1 TO 4
```

DØ# = DI\$(I)

P = VAL(DØ#) - 47

X = VAL(STR(DI\$(I), 4))

0011 0010 .. 36 82 04 2. 31 90 00 3. 37 82 0A

1. P = 764 36 = BIN 54 ENTRY FOR DRIVES IN DØ#( )

2. P = 2 no drive 3. P = 8

```
1115 IF P < 1 THEN STOP ERROR
```

1ST BYTE OF DI\$(I) MUST BE AT LEAST 30 (BINARY 48)

```
1116 IF P > 8 THEN ILLEGAL SW SET
```

1ST BYTE "

MAXIMUM 37 (BINARY 55)

```
1120 IF STR(DØ#, 2, 1) = HEX(86) THEN DRIVE UNCONFIGURED
```

```
1130 IF STR(DØ#, 1, 2) = HEX(31, 96) THEN DO NO DRIVE
```

```
1140 IF STR(DØ#, 2, 1) = HEX(96) THEN BAD DRIVE
```

```
1145 IF NI > 14 OR NZ > 14 THEN # OF ADDRESSES > 14 ON MASTER OR SLAVE
```

```
1150 CONVERT STR(DØ#(P), 3) TO Z
ERROR Z = Ø
```

1. Z = 64 3. Z = 140

DØ#(7), 3 = 640

20 DIM DØ#(16) 40

```
1160 IF Z/10 = INT(Z/10) THEN Z = 38912 ELSE Z = 65024 Z ASSIGNED SECTOR SIZE
```

```
1165 IF STR(DØ#, 1, 1) = HEX(32) THEN DO FOR 10M REM
```

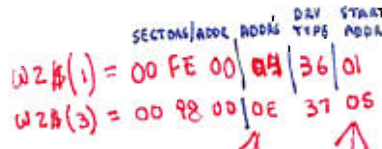
```
1170 W(I) = Z * P(P)
```

TOTAL SECTORS FOR CURRENT DRIVE SURFACES

START ADDR

WZØ(I) = BIN(Z, 3) & BIN(P(P)) & STR(DØ#(I), 1) & STR(DØ#(I), 5, 1)

WØ(I) = P 1. 7 3. 8



```
1175 NEXT I
```

```
1176 DIM C1Ø(4) 1, C2Ø(4) 6, C3Ø(4) 2
```

FOR I = 1 TO 4

KØ = STR(DIØ(I), 5, 1) 1. KØ = 01 2. 00 3. 05 4. 00 START ADDR

IF KØ = HEX(00) THEN KØ = "Z" 2. KØ = 5A 4. KØ = 5A

C1Ø(I) = KØ 1. 01 2. 5A 3. 05 4. 5A

NEXT I

MATSORT C1Ø() TO C2Ø(), C3Ø()

C1Ø() = 01 5A 05 5A

C2Ø() = 67 89 05 00 69 89 05 00 68 89 05 00 6A

C3Ø() = 00 01 00 03 00 02 00 04

FOR I = 1 TO 4

CZØ(I) = DIØ(VAL(C3Ø(I), 2))

1. CZØ() = 36 82 04 01 01 04 05 00 68 89 05 00 6A 89 05 00 31 96 00 02 31 96 00 00

DIØ = CZØ

```
1180 RETURN
```

```
210 REM
```

```
250 REM DISPLAY INFO ON DRIVES
```

```

FOR I = 1 TO 32
  IF I = 17 THEN STR(D#, 2, 1) = OR HEX (04)  D# = D20
  GOSUB 120 (STR(E#(I), 2, 1))
290 DEFFN 120 (G#)
  IF STR(G#(I), 3, 1) = HEX(04) THEN RETURN =,00 2.00
  MATSEARCH H#( ), = G# TO P#  H# = 0-F 0-F
  IF P# = HEX(00 00) THEN RETURN
  P, P = VAL(P#, 2)  ∴ P, P = 16 2.7
300 P# = P(P)  ∴ P# = 1 2. P# = 4
  ON I GOTO 310
  310 PRINT BOX
  320 PRINT "DISKETTE DRIVE" 64 MB
  IF STR(E#(I), 17, 1) = HEX(00) THEN 325
  325 W1 = VAL(STR(E#(I), 6, 3), 3)  ∴ W1 = 4160 2. 65024
  IF I > 1 THEN 326
  IF W1 = 4160 ∴ 1.2 M
  IF W1 = 1280 ∴ 360K
  GOTO 360
  360 PRINT
  FOR P1 = 1 TO P#
    D1# = STR(D#, 1, 2) & H#(I + P1 - 1)  ∴ D20 2. D21
    PRINT USING 385, VAL(STR(E#(I + P1 - 1), 6, 3)); D1#
    IF P1 = 5 OR P1 = 10 THEN DO
  NEXT P1
  370 PRINT
  370 I = I + P# - 1  I = 1
  IF W2 = 4 THEN I = 31
  380 RETURN
250 IF I = 1 OR I = 16 OR I = 31 THEN PRINT
  NEXT I

```

STANDARD ADDR / DISK TYPE 64M = 4 140 = 10

W2 = W2 + 1 : IF R# > "32" THEN 305  
 305 P1 = VAL(D1#(W2)) - 47 P1 = 7  
 P# = MIN(14, VAL(STR(D1#(W2), 6))) P# = 4  
 IF P# > 0 THEN 310

326 IF I = 32 THEN 330 - LAST ADDR  
 IF G# = "2" THEN 350 NO DRIVE  
 IF R# > "32" THEN 330 R# = 40  
 330 IF I < 32 THEN 350  
 350 IF P# > 0 THEN # ADDR P# = 4  
 352 IF G# = "2" MUST BE DRIVE 1  
 354 IF P# = 0 THEN 370

64M 1-4

2.65024 ON D21

4010 W DRIVE # W2 # WINGS W (MAX SECTORS / WING) W# (SECTORS / SURF) W4# START #  
 W5# SURFACE ON DRIVE M/S



# APPLYING CASE 5

LOAD "C:\DSAPPLY" 9020,9999 BEG 9005

05 INIT(00)W3# = 000000

GOSUB 8250

250 INIT(00)R#( )

CASES 1098-1103  
1099 OR 01

DATALOADBAT#1, (A,A)R#( ): READ CONFIG DATA

252 X=0

IF STR(R#( ),11,3) <> "TBO" THEN X=1: = TBO

IF R#(3) = HEX(00-00) THEN X=1:

9005 IF X=0 THEN GOSUB 9020

9020 REM

9025 BIN(STR(R#( ),1))=W W=1 FOR DRIVE 1

9030 PRINT HEADING

FOR X=1-16

PRINT AT(3+X,1); HEXOF(R#(X)) PRINT 1<sup>ST</sup> 16 BYTES, PRINTS

NEXT X

9040 I = VAL(STR(R#( ),16)) I=8 LAST BYTE 1<sup>ST</sup> LINE

IF I > 0 THEN STR(R#( ),16,4) = CH(I) CS(I) = 0EFF0806 REM .APPLY 4 CONT BYTES

9050 FOR X=3 TO 16 FE0000 SECTORS FOR EACH ADDR

IF STR(R#(X),4,3) > HEX(00 00 00) THEN STR(R#(X),3) = W3# W3# = 000000

ADD C(STR(W3#,3), STR(R#(X),4,3))

NEXT X ADDR FIELDS

9060 K# = K# + 1 ADDSTR(R#( ),X,1): NEXT X 256 TIMES AT END HEXPRINT K# = 21

3216 8 4 21  
10 0001

DO YOU WANT TO APPLY N

9005 NEXT W W=2

INIT(00)W3# : GOSUB 8250

8250 INIT(00)R#( )

DATALOADBAT#1, (A,A)R#( ) ALL 00s

8252 X=0

IF STR(R#( ),11,3) <> "TBO" THEN X=1 ✓

IF R#(3) = HEX(00-00) THEN X=1

RETURN

9005 IF X=0 THEN 9020

NEXT W 3







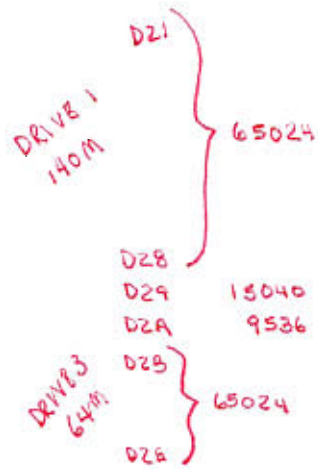
## DEFAULTS

CASE3 1/28/94

Proposed DS Address Assignments  
 2 Winchester with sectors available

CASE 5 CREATED

Master disk Address		Catalog Maximum	Slave disk Address		Catalog Maximum
D21	1	38912	D61	0	0
D22	1	38912	D62	0	0
D23	1	38912	D63	0	0
D24	1	38912	D64	0	0
D25	1	38912	D65	0	0
D26	1	38912	D66	0	0
D27	1	38912	D67	0	0
D28	1	38912	D68	0	0
D2A	1	38912	D69	0	0
D2B	3	65024	D6A	0	0
D2C	3	65024	D6C	0	0
D2D	3	65024	D6D	0	0
D2E	3	65024	D6E	0	0





2/1/93 ADDED LINE 617 TO INSURE EITHER DOS OR 2200 FORMAT WAS SELECTED IF USING 5 1/4" FLOPPY.

NEW LINE 617 IF A2# = "2" THEN 620: IF A2# = "1" THEN 620: GOTO 615

IF FORMATTING AN ADDRESS WHICH HAD ALREADY BEEN FORMATTED AND YOU ANSWER No TO WHETHER OR NOT TO CONTINUE FORMAT, PROGRAM WOULD TAKE YOU BACK TO MOUNT PLATTER SCREEN. CHANGE PROGRAM TO TAKE YOU BACK TO THE ADDRESS SCREEN IF RESPONDED WITH "N".

CHANGE LINE 780 780::: IF K# = "N" THEN 510:  
WAS IF K# = "N" THEN 610:

2/5/93 WHEN FORMAT FAILED, ERROR DID NOT DISPLAY, WENT BACK TO EITHER DOS OR 2200 FORMAT SELECTION IF FLOPPY OR MOUNT DISK. THEN AFTER KEY RETURN ERROR SHOWS IN LOWER LEFT CORNER.

MADE LINE 9040 TO 9050 ERROR GOSUB RETURN LINE

COPIED LINE 7010 TO 9040 PRINTS ERROR

SPLIT LINE 610 CREATING LINE 612

ON 610 ADDED 610 # CLOSE #1: IF M <> 0 THEN 612:

ON 612 ADDED 612 PRINT AT (9,0,80): M2 = 1: A2# = "1": IF A1# = "D10" OR A1# = "D20" OR A1# = "D30" THEN 615: GOTO 620

2/8/93 WHEN USING DOS FORMAT WITH FLOPPY WOULD NOT COME BACK - SAY "FORMAT COMPLETED". CAME BACK TO MAIN MENU.

CHANGED LINE 813 TO GOTO LINE 820 IF SUCCESSFUL, NOT LINE 815.

813 IF STR(G#, 6, 3) = HEX(00 00 00) THEN (815) 820

3/12/93 CFORMAT DOES NOT RECOGNIZE IF THE DISKETTE HAS <sup>INDEXED</sup> DATA ON IT WHEN A DOS FORMAT IS CHOSEN.

Fix: DELETE LINE 705 IF A2#="2" THEN 795

SOMETIMES LONG MESSAGES ARE NOT CLEARED AT THE BOTTOM OF SCREEN.

Fix: 7010 PRINT AT (23,0,33); HEX(0E); STR(M#); HBX(01): REM-DISPLAY MESSAGE  
7010 PRINT AT (23,0,50);

LINE 110. CHANGED VERSION TO 1.07.00 FROM 1.06.00.



# OLD FORMAT 3.4

# NEW FORMAT

FORMAT COMPLETED, SCRATCH  
MSG  
NO ~~MSG~~ WENT BACK TO ADDR ✓

FORMAT COMPLETED  
NO MSG, BACK TO ADDR

OK  
NO MSG, BACK TO ADDR

OK  
NO MSG

VLSI / 2275

CS/2200

DOS  
WINCHESTER

386 / DS 1.2 R3.F

CS/2200

• DOS •  
WINC  
TURBO / 2275

CS/2200  
~~WINC~~

• DOS •

TURBO / DS 1-2 3.5

CS/2200

• DOS •

SCSI

FORMAT COMPLETED, SCR

FORMAT COMPLETED, SCRATCH  
FORMAT COMPLETED

FORMAT COMPLETE

FORMAT COMPLETED  
FORMAT COMPLETE

OK  
~~FORMAT~~  
OK

GOOD

GOOD

GOOD



```

F=1: KEYIN K$: IF K$=HEX(7E)OR K$=HEX(7F)THEN 1880: IF K$<>HEX(0D)THEN 650
700 REM %GET & DISPLAY INDEX INFORMATION (IF INDEX SECTORS > 0)
710 IF M2=-1THEN SELECT #1(A1$): ERROR GOSUB '255: GOTO 610: REM - SELECT DISK h
hh
725 V$(1)="24": V$(2)=" ": V$(3)="0": REM - INITIALIZE THE INDEX INFORMATION FIE
LDS: X=0
730 %OPEN 732,#1: ERROR GOSUB '255: GOTO 610: REM - "HOG" THE DISK
731 GOTO 740: REM - SKIP THE "CAN'T %OPEN" MESSAGE
732 ="Drive Already In Use or Not Ready": PRINT HEX(07): GOTO 610: REM - DISPL
AY MESSAGE IF CAN'T %OPEN DRIVE
740 IF B0$="340"THEN 760: G$(11)=HEX(FF): %BITSETDISKTYPE#1(0200030F122206000700
70A04000870B,G$(1)): REM - SET THE DRIVE TYPE (DPU TYPE)
750 IF STR(G$(1),6,3)=HEX(00 00 00)THEN 752: M$="Drive Not Responding": PRINT HEX
(07): GOTO 610
752 IF G$(11)=HEX(00)THEN 760: IF G$(11)=HEX(0C)THEN 760: IF G$(11)=HEX(FF)THEN
PRINT AT(6,0,80);"No disk controller at address": GOTO 565
760 DATA LOAD BA T#1,(0)G$(1): ERROR GOSUB '255: IF M<>93THEN 610: M$=" ": GOTO 8
00: REM - GET FIRST INDEX SECTOR. FORMAT DISK IF FORMAT ERROR. REPORT ALL OTHE
R ERRORS
765 IF STR(S$(1),2,1)=HEX(00)THEN 800: REM - IF INDEX SECTORS = 00000, FORMAT IT
770 FOR I=1TO 3: IF I=1THEN CONVERT VAL(STR(S$(1),2,1))TO V$(I),(####): ELSE CO
NVERT VAL(S$(1),2)-1TO V$(I),(####): IF V$(I)="00000"THEN V$(I)="0": ELSE V$(I)
=STR(V$(I),POS(V$(I)<>"0")): NEXT I: REM - CHANGE THE 2-BYTE VALUES TO STRINGS
775 PRINT AT(13,27);"##### 8p10A 8p0·#aa#·p";AT(15,29);"Index Sectors = ";V$(
1);AT(16,29);"End Cat. Area = ";V$(3);AT(17,29);"Current End = ";V$(2): REM -
SHOW HIM THE CURRENT INDEX
776 IF STR(S$(1),1)<>HEX(00)THEN PRINT AT(15,51);" "
780 PRINT HEX(07): GOSUB '201("Are you sure (Y/N)?"): M$=" ": GOSUB '204: IF K$=
"N"THEN 510: REM - GIVE HIM THE FINAL WARNING
795 REM %TRASH THE SUCKER!
800 M$=HEX(02 04 04 00 0E)&"(Formatting)"&HEX(02 04 00 00 0E): GOSUB '201(M$): R
EM - TELL HIM THE NASTY THINGS WE'RE DOING
810 IF A2$="2"THEN 812: %FORMATDISK T#1: ERROR GOSUB '255: GOTO 610: REM - FINAL
L %FORMAT THE DISK
811 GOTO 820
812 REM ! PC FORMAT: SELECT #1(A1$): G$=ALL(20): %BIT#1(0600070070A0400288D07040
01306A10680240018B67,G$)
813 IF STR(G$,6,3)=HEX(00 00 00)THEN 820
814 IF STR(G$,6,3)<>HEX(00 00 00)THEN DO: PRINT HEX(07); M$=HEX(02 04 02 0E)&"D
isk Error - Press any Key "&HEX(0F): GOSUB '201(M$): KEYIN K$: M$=ALL(" "): ENDD
O: GOTO 610
815 GOTO 510
820 GOSUB '201("Format Completed"): REM - WE MADE IT!
822 F=2
825 REM %GET NEW INDEX PARAMETERS & SCRATCH THE DISK
830 PRINT AT(6,0,80);HEX(0E);"Enter new index information and press RUN:": REM -
CHANGE THE PROMPT
835 IF STR(S$(1),1)=HEX(01)THEN V$(2)="NEW": ELSE V$(2)="OLD": REM - INITIALIZE
THE STRUCTURE FIELD
840 PRINT AT(13,0,400);HEX(02 04 02 04 0F);AT(13,29);"8A 8p10A 8p0·#aa#·p";AT(
15,29);"Index Sectors = ";HEX(0E);STR(V$(1));HEX(0F);AT(16,29);"End Cat. Area =
";HEX(0E);STR(V$(3));HEX(0F);AT(17,29);"Structure = ";HEX(0E);STR(V$(2),,3);
HEX(02 04 02 00 0F);" (OLD or NEW)": REM - DISPLAY THE FIELDS
845 PRINT AT(21,52);"RETURN - Next Field";AT(22,52);"RUN - Accept parameters"
: REM - ADDITIONAL KEY DEFINITIONS
850 K0=1: REM - RESET FIELD COUNTER
855 ON K0GOSUB 875,880,885: REM - PROCESS THE CURRENT FIELD
858 GOSUB '201(" "): REM - ERASE THE MESSAGE (IF ANY)
860 POS(HEX(82 7E 7F)=K$)GOTO 900,520,520: REM - RUN , FN / TAB
861 K0=K0+1: IF K0>3THEN K0=1: GOTO 855: REM - RETURN PRESSED, ADVANCE TO NEXT F
IELD
875 GOSUB '202(15,45,V$(1),5,"#",HEX(7F 7E)): V$(1)=10$(1): RETURN: REM - ACCEPT
INDEX SECTORS
880 GOSUB '202(16,45,V$(3),5,"#",HEX(7F 7E)): V$(3)=10$(1): RETURN: REM - ACCEPT
END OF CATALOGED AREA
885 GOSUB '202(17,45,V$(2),3,"#&" HEX(7F 7E)): V$(2)=10$(1): %STRAN(V$(2),"AaBbCcD

```



```

deefBgHhIiJjKkLlHhMnOoPpQqRrSsTtUuvVwWxYyZz"JR: IF V#(2)="OLD"OR V#(2)="NEW"TH
EN RETURN: GOSUB '201("Index Structure Must be 'OLD' or 'NEW'"): PRINT HEX(07):
GOTO 885: REM - ACCEPT INDEX STRUCTURE
900 GOSUB '201("Creating Disk Index"): REM - TELL HIM WHAT WE'RE UP TO
910 CONVERT V#(1)TO V1: ERROR PRINT HEX(07): GOSUB '201("Index Sectors may not b
e blank"): GOTO 850: REM - CHANGE INDEX SECTORS INTO A NUMERIC VALUE
912 IF V1>=1AND V1<=255THEN 915: PRINT HEX(07): GOSUB '201("Index Sectors must b
e 1 to 255"): GOTO 850: REM - VALIDATE INDEX SECTORS VALUE
915 CONVERT V#(3)TO V3: ERROR PRINT HEX(07): GOSUB '201("End Cat. Area may not b
e blank"): GOTO 850: REM - CHANGE END OF CATALOGED AREA INTO A NUMERIC VALUE
917 IF V1<=V3THEN 920: PRINT HEX(07): GOSUB '201("End Cat. Area less than Index
Sectors"): GOTO 850
920 IF V#(2)="OLD"THEN SCRATCH DISK T#1,LS=V1,END=V3: ERROR GOSUB '255: GOSUB '2
01(M#): GOTO 850: REM - BUILD THE OLD INDEX STRUCTURE & HANDLE ERROR
930 IF V#(2)="NEW"THEN SCRATCH DISK 'T#1,LS=V1,END=V3: ERROR GOSUB '255: GOSUB '
201(M#): GOTO 850: REM - BUILD THE NEW INDEX STRUCTURE & HANDLE ERROR
940 GOSUB '201("Index Created - Press Any Key to Exit"): PRINT HEX(07): KEYIN K#
: REM - LET HIM KNOW WE'RE FINISHED
999 #CLOSE #1: GOTO 520: REM - FORMAT/SCRATCH COMPLETED. "UNHDS" THE DISK & GO
TO THE DISK ADDRESS MENU
1870 DEFFN'126
1880 DEFFN'127: GOSUB '201("FN/TAB keyed")
1890 IF F=1THEN 510
2000 COM CLEAR: GOSUB '201("Returning to System Menu"): #PSTAT=" ": LOAD DC T"
#MENU": END: REM - THAT'S ALL SHE WROTE
5025 DEFFN'100(S#): IF S#="340"THEN 5035: #TRAN(S#,"AaBbCcDdEeFf"IR: IF POS("DB3
"=S#)>0AND POS("123567"=STR(S#,2))>0AND (VER(STR(S#,3),"H"))>0THEN 5030: Q#="I": R
ETURN
5030 IF POS("3B"=S#)=0OR POS("123"=STR(S#,2))=0OR STR(S#,3)<>"0"THEN 5035: IF ST
R(S#,1)="3"THEN STR(S#,3)="1": STR(S#,1)="D"
5035 Q#=" ": RETURN
5040 REM % Get disk status
5050 G#=#ALL(20): STR(B#,2,1)=STR(B#,2,1)AND HEX(33): SELECT #2(B#)
5060 G10GETDISKTYPE#2(0200030F12220600070070A04000870B,G#(1)
5090 T#=#(1): T=POS(HEX(C0 D0)=T#): IF T=0THEN 5140: IF T=2THEN GOSUB 5150
5100 GN T6GOSUB 5240,5250,5240,5230: PRINT K1#
5110 D#=#(1)
5130 RETURN
5140 PRINT "Unknown disk type check connector": D#(1)=" ": RETURN
5150 REM % Check for DS cabinet
5160 G#=#HEX(30): STR(G#,2,7)=ALL(00): #G10STATUSREQUEST#2(0E140F0012E20600070070
A0400288D070406A106816400087051A00C340,G#)G#;STR(E#,VAL(STR(G#,5,1)))
5170 T=VAL(E#)-47
5180 G1#=#STR(G#,8,1)AND HEX(10): IF G1#=#HEX(00)THEN 5190: K1#="Disk unavailable"
: RETURN
5190 IF STR(G#,8,1)<>HEX(00)THEN 5200: IF STR(E#,1,1)=HEX(33)THEN 5220: IF I=32#
ND STR(E#,1,2)=HEX(45 33)THEN 5210
5200 REM "specified disk is not a DS": RETURN
5210 STR(E#,1,2)=HEX(33 45)
5220 RETURN
5230 K1#="DS cabinet": RETURN
5240 K1#="2275 type": RETURN
5250 K1#="Phoenix type": RETURN
5260 K1#="2270 type": RETURN
6000 REM % GET DISK STATUS
7000 DEFFN'201(M#): REM % DISPLAY M# IN LOWER LEFT CORNER OF THE DISPLAY
7010 PRINT AT(23,0,50);HEX(0E);STR(M#);HEX(01): REM - DISPLAY MESSAGE
7020 RETURN
70 DEFFN'202(I1,I2,I0#(1),I3,I0#,I1#): REM % ACCEPT FIELD (I1,I2=ROW,COL I0#(1
)=DATA I3=LEN I0#=#VER MASK I1#=#EXIT SFKEYS)
7040 MAT REDIM I0#(1)I3: REM - CHANGE LENGTH OF FIELD TO LENGTH SPECIFIED
7050 PRINT HEX(02 04 02 04 0E): REM - SET ATTRIBUTES TO BRIGHT/UNDERSCORE
7060 I0=#: REM - INITIALIZE CURSOR POSITION
7070 PRINT HEX(06);AT(I1,I2);STR(I0#(1)): REM - DISPLAY THE FIELD
7080 PRINT AT(I1,I2+I0-1);HEX(02 05 0F): REM - POSITION CURSOR
7090 KEYIN K#,7200: REM - WAIT FOR A KEY. BRANCH IF SF KEY PRESSED

```

```

7110 ON POS(HEX(20 08 0D 82)=K#)GOTO 7130,7150,7180,7190: REM - BRANCH IF SPACE,
BACKSPACE, RETURN OR RUN/EXEC
7120 IF VER(K#,I0#)<> THEN 7080: REM - IF NOT A VALID KEY, WAIT FOR ANOTHER
7130 IF I0#<HEX(00) THEN 7080: STR(I0#(1),I0,1)=K#: PRINT K#: REM - IF ACCEPTING
CHARACTERS, PUT THE CHARACTER INTO THE FIELD & PRINT IT
7140 IF I0<13 THEN I0=I0+1: GOTO 7080: REM - INCREMENT CURSOR POSITION & UPDATE I
T
7150 I0=I0-1: GOTO 7080: REM - BACKSPACE PRESSED; GET ANOTHER KEY IF AT LEFTMOST C
OLUMN
7160 IF I0<13 OR STR(I0#(1),I0,1)="" THEN I0=I0-1: REM - DON'T DECREMENT CURSOR P
OSITION IF AT RIGHTMOST POSITION & IT'S NON-BLANK
7170 STR(I0#(1),I0,1)=" ": GOTO 7070: REM - REMOVE THE CHARACTER & UPDATE THE CR
T
7180 K#=HEX(20): GOTO 7280: REM - RETURN PRESSED; TRANSLATE IT TO HEX(20) & EXIT
7190 GOTO 7280: REM - RUN/EXEC PRESSED; EXIT
7200 PRINT HEX(06): REM - SF KEY PRESSED; SHUT OFF THE CURSOR
7210 IF POS(11#<K#)<> THEN 7280: REM - IF AN EXIT KEY, EXIT
7220 IF I0#<HEX(00) THEN ON POS(HEX(48 49 4A 4C 4D)=K#)GOTO 7230,7240,7250,7260,
7270: GOTO 7080: REM - IF MODIFIABLE, BRANCH TO ERASE, DELETE, INSERT, RIGHT OR
LEFT
7230 STR(I0#(1),I0)="" : GOTO 7070: REM - ERASE PRESSED; DO IT & UPDATE THE CRT
7240 IF I0<13 THEN STR(I0#(1),I0)=STR(I0#(1),I0+1): STR(I0#(1),13,1)="" : GOTO 70
70: REM - DELETE PRESSED; DO IT & UPDATE THE CRT
7250 IF I0<13 THEN MAT COPY -STR(I0#(1),I0,13-I0) TO -STR(I0#(1),I0+1): STR(I0#(1)
,I0,1)="" : GOTO 7070: REM - INSERT PRESSED; DO IT & UPDATE THE CRT
7260 IF I0<13 THEN I0=I0+1: GOTO 7070: REM - RIGHT PRESSED; MOVE CURSOR & REPOSITI
ON IT
7270 IF I0>1 THEN I0=I0-1: GOTO 7070: REM - LEFT PRESSED; MOVE CURSOR & REPOSITIO
N IT
7280 PRINT HEX(02 04 02 00 0F): REM - SET THE ATTRIBUTES TO BRIGHT (NORMAL)
7290 MAT REDIM I0#(1)80: REM - RESET THE FIELD LENGTH TO ITS MAXIMUM
7300 RETURN
7310 DEFFN'204: REM % RETURN K# = "Y" or "N"
7320 KEYIN K#
7325 IF K#=HEX(7E) OR K#=HEX(7F) THEN K#="N"
7330 #TRAN(K#,"YyNn"): IF K#<>"Y" AND K#<>"N" THEN 7320
7340 RETURN
9000 DEFFN'255: REM % ERROR HANDLER (SETS M#<=ERROR DESCRIPTION & BEEPS)
9005 M=ERR: REM - GET THE ERROR NUMBER
9010 M#="" : REM - INITIALIZE MESSAGE
9015 IF M<90 THEN 9025: RESTORE LINE 9020,M-89: READ M#: REM - GET MESSAGE IF DISK
ERROR
9020 DATA "Disk Hardware Error","Disk Drive Not Ready","Disk Drive Time-Out","Di
sk Format Error","Disk Format Key Engaged": REM - DISK ERRORS
9021 DATA "Disk Seek Error or Platter Protected","Disk CRC Error","Disk LRC Erro
r","Bad Sector Address/Platter Not Mounted","Verify Error": REM - DISK ERRORS (C
ONT.)
9025 IF M=48 THEN M#="Illegal Device Specification"
9030 IF M#<>" " THEN 9040: M#="Unexpected Error ### Occurred": CONVERT M TO STR(M#
,I0,3),(###): REM - CATCH-ALL
9040 PRINT AT(23,0,33);HEX(0E);STR(M#);HEX(01): REM - DISPLAY MESSAGE
9050 PRINT HEX(07): RETURN

```



# NEWFORM D11 ★

260 DIM V\$(3) 8: REM

310 J\$ = \$PSTAT(1)

761 IF STR(S\$( ), 1, 1) <> HEX(02) THEN 765: IF STR(S\$( ), 2, 2) = HEX(0000) THEN 800:

REM - IF 3 BYTE INDEX = 00000 THEN FORMAT IT

762 CONVERT VAL(STR(S\$( ), 2, 2)) TO V\$(1), (#####):

CONVERT VAL(STR(S\$( ), 4, 3)) TO V\$(2), (#####):

CONVERT VAL(STR(S\$( ), 7, 3)) TO V\$(3), (#####)

763 PRINT AT(14, 29); INDEX Type = 3 Bytes: GOTO 775

776 IF STR(S\$( ), 1) = HEX(01) THEN PRINT AT(15, 51); " " : IF STR(S\$( ), 1) = HEX(02) THEN PRINT AT(15, 51); " 8"

840 PRINT AT ..... (OLD or NEW) ~~OR~~

841 IF STR(J\$, 9, 1) <> "T" THEN 845: PRINT AT(17, 64); " or 3": REM - IF TURBO ALLOW 3 BYTE OPTION  
OR STR(S\$( ), 3) = "0" (KEY TRI)  
IF T < > 15 AND Z\$ < "3F" THEN 845:

~~885 GOSUB '202: ATRAN(V\$(2), "Aa-Zz 3" THEN RETURN: GOSUB '201("INDEX STRUCTURE MUST BE 'OLD' OR 'NEW' OR IF TURBO OPTIONALLY '3'")~~

~~880 GOSUB '202(15, 45, V\$(1), 8, "#", HEX(7F 7E)): V\$(1) = I\$(1): RETURN: REM~~

~~885 GOSUB '202(17, 45, V\$(2), 3, "A", HEX(7F 7E)):: IF V\$(2) = "OLD" OR V\$(2) = "NEW" OR V\$(2) = "TRI" THEN RETURN~~

~~8 GOSUB '201("INDEX STRUCTURE MUST BE 'OLD' OR 'NEW' OR IF TURBO OPTIONALLY 'TRI'"):: REM~~

4/23/93

D11 DISKTYPE

NEWFORM2  
NEWFORM3

```

827 GOSUB'300: REM - CHECK DISK TYPE
8000 DEFFN'300: REM %↑. GET DISK STATUS
8010 275 DIM Z# 2, X# 32, X$(1) 32: REM - DISK TYPE VARIABLE (3 BYTE CHECK) (320)
8020 B$=D0$: A$=ALL(20): STR(B, 2, 1) = STR(B#, 2, 1) AND HEX(33): SELECT # 2 < B# >
8030 $G10GETDISKTYPE# 2 (0200030F12220600070070A04000870B, G$( ))
8040 T# = G$(11): T = POS(HEX(C0D0) = T#): IF 2 <> 2 THEN RETURN
8050 REM %↑. CHECK FOR DS CABINET
8060 G# = HEX(30): STR(G#, 2, 7) = ALL(00): $G10STATUSREQUEST# 2 (0B140F0012E20600070070
A040D288D070406A106816400087051A00C340, G#) G#; STR(X#, VAL(STR(G#, 5, 1)))
ERROR GOSUB'255
8070 T = VAL(X#) - 47: IF T <> 4 THEN RETURN: REM - RETURN IF NOT A DS
8080 STR(G#, 2, 7) = ALL(00): G10READ
8090 IF STR(X#, 2, 1) = HEX(00) THEN STR(E#, 18, 1) = HEX(00): G1# = STR(G#, 8, 1) AND HEX(10):
IF G1# > HEX(00) THEN 8120
8100 IF STR(G#, 8, 1) <> HEX(00) THEN 8130: X$(1) = X#
8110 Z# = STR(X$(1), 4, 2): REM - Z# = DS FROM REVISION
8115 RETURN
8120 GOSUB'255 ("DISK UNAVAILABLE")
8125 RETURN
8130 GOSUB'255 ("SPECIFIED DISK IS NOT A DS")
8135 RETURN

```

T#2 NOT A SCSI DS

```

87: % REVISION = 26 APRIL 1993 (MEB) 3-BYTE RECOGNITION ADDED.
110 V#8: V# = "02.00.00": REM - VERSION NUMBER
520 $CLOS#1: PRINT TAB(20); "(c) COPR. WANG LABS, INC. 1993: PRINT TAB(31); "REL "; V#

```





NBWF Rms  
DISKTYPE  
NEWFORM3

3 BYTE SHOWS 1 MORE SECTOR THAN LST FOR END CAT AREA +  
CURRENT END

2 BYTE MATCHES  
3 BYTE SCRATCH TAKES LONG TIME INDEX = 26000 END CAT 65601

885 GOSUB '202(17,45,V\$(2),3,"A",HEX(7F 7E)):V\$(2)=I\$(1):\$TRAN(V\$(2),  
"Aa-Zz")R

886 IF V\$(2)="OLD" OR V\$(2)="NEW" THEN RETURN: IF V\$(2)="TRI" AND STR(J\$,9,1)="T"  
AND STR(S\$,3)>0 THEN RETURN

887 IF STR(J\$,9,1)="T" THEN 888: GOSUB '201("Index Structure Must be 'OLD' or  
'NEW'"): PRINT HEX(07): GOTO 885: REM - ACCEPT INDEX STRUCTURE

888 GOSUB '201("Index Structure Must be 'OLD', 'NEW', or 'TRI' for 3 Byte"):  
PRINT HEX(07): GOTO 885: REM - ACCEPT INDEX STRUCTURE FOR TURBO

911 IF V\$(2)="TRI" AND STR(J\$,9,1)="T" AND STR(S\$,3)>0 AND Z\$>="3F" THEN 913  
AND STR(S\$,3)>0 AND T=-15

912 IF V\$(2)="TRI" AND STR(J\$,9,1)="T" THEN 913: IF V1>=1 AND V1<=255 THEN 915:  
PRINT HEX(07): GOSUB '201("Index Sectors must be from 1 to 255"): GOTO 850:  
REM - VALIDATE INDEX SECTORS VALUE

913 IF V1>=1 AND V1<=65535 THEN 915: PRINT HEX(07): GOSUB '201("Index Sectors  
must be from 1 to 65535 for 3 Byte"): GOTO 850: REM - VALIDATE INDEX SECTORS  
VALUE FOR 3 BYTE

917 IF V1<=V3 THEN 918: PRINT HEX(07): GOSUB '201("End Cat. Area less than Index Sectors"):  
GOTO 850

918 IF V\$(2)="TRI" AND STR(J\$,9,1)="T" THEN 920: IF V3>=V1 AND V3<=65535  
THEN 920: PRINT HEX(07): GOSUB '201("End Cat. Area must be less than  
65535"): GOTO 850: REM - VALIDATE 2 BYTE END CAT. AREA

933 IF STR(S\$,3)="0" THEN 940: IF T<>-15 AND Z\$<"3F" THEN 940:

935 IF V\$(2)<>"TRI" AND STR(J\$,9,1)<>"T" THEN 940: SELECT 3 ON: SCRATCH DISK & T#1,  
LS=V1, END=V3: ERROR GOSUB '255: GOSUB '201(M\$): GOTO 850: REM - BUILD THE  
3 BYTE INDEX: STRUCTURE & HANDLE ERROR

934 IF <256 THEN 935: GOSUB '201("BE PATIENT, LARGE INDICES TAKE TIME")



# NEWFORM3

10 % @FORMAT 03/20/90 -- (c) Copr. Wang Laboratories, Inc. 1986  
 20 % (c) Copr. Wang Laboratories, Inc. 1986

40 % Program Name = @FORMAT  
 50 % Author = Steve McGarry  
 60 % Date Written = 22 April 1986  
 70 % Last Revised = 20 March 1990 (TBC)  
 80 % Last Revised = 10 February 1993 (MEB)  
 85 % Last Revised = 12 March 1993 (MEB)  
 87 % Revised = 26 April 1993 (MEB) 3 Byte Recognition added.

```

95 DIM A1$3,A2$1,S$3,O$1
100 REM %VARIABLE DEFINITIONS
110 DIM V$9: V$="02,00,00": REM - VERSION NUMBER
120 DIM D$3,K,P#1: REM - DEVICE/PLATTER ADDRESS VARIABLES
130 DIM E$(40)1,G$(15)1: REM - STATUS/%GID VARIABLES
160 DIM K$1: REM - KEYIN BYTE
170 DIM M$40: REM - MESSAGE ('201)
180 DIM M: REM - ERROR HANDLER WORK VARIABLE
190 DIM IO$(1)80,IO$1,I1$80,IO,I1,I2,I3: REM - VARIABLES FOR ACCEPTING A FIELD
200 DIM I,J,JO: REM - LOOP INDEXES
210 DIM N: REM - GENERAL WORK VARIABLE
220 DIM M2,C,R: REM - DISK MENU VARIABLES ('210)
230 DIM D,D1: REM - DISK SIZE, DESCRIPTION NUMBER ('222)
240 DIM D0$3: REM - "OTHER" DISK ADDRESS
250 DIM S$(128)2: REM - INDEX SECTOR BUFFER
260 DIM V$(3)6: REM - INDEX SECTOR DATA CONVERSION VARIABLES
270 DIM V1,V3: REM - INDEX INFORMATION VALUES
280 DIM Z$2,X$32,X$(1)32: REM - DISK TYPE CHECK VARIABLES FOR 3 BYTE
290 DIM G1$17,E1$32,P$1,G1$(32)3,B$3,G2$1: P$=HEX(10): REM VARIABLES FOR GETTING
  DISK STATUS
300 REM %1.Initialize everything: REM %1st display
310 J$=$PSTAT(1)
510 SELECT PRINT /005(80): PRINT HEX(02 0D 0C 03 0F)
520 %CLOSE #1: PRINT HEX(03 06 0E);TAB(15);"SOFTWARE FORMATTABLE DISK PLATTER FO
  RMAT UTILITY": PRINT TAB(20);"(c) Copr. Wang Laboratories, Inc. 1993": PRINT TAB
  (31);"Release ";V$
540 PRINT AT(22,52);"RETURN - Proceed";AT(23,52);"FN/TAB - Previous Menu";HEX(01
  )
560 REM %Get Disk Address
565 PRINT AT(4,0);HEX(02 04 02 04 0F);"Enter platter address: ";HEX(0E);D0$;HEX(
  02 04 02 00 0F)
570 F=0: PRINT AT(4,22);: LINPUT HEX(0E),-D0$
572 GOSUB '201(" ")
580 A1$=D0$: $TRAN(A1$,"BbDd")R: B$=A1$: GOSUB '100(A1$): IF Q$=" "THEN 610
585 GOSUB '201("Platter address must be 3 hex digits"): PRINT HEX(07): GOTO 570
600 REM %Show Platter information: REM .Unhog disk and show 2nd screen
610 %CLOSE #1: IF M<>0THEN 612: PRINT HEX(03 06 0E);TAB(15);"SOFTWARE FORMATTABL
  E DISK PLATTER FORMAT UTILITY": PRINT TAB(30);"Platter Information": PRINT AT(4,
  0);"Platter address = ";D0$,
612 PRINT AT(9,0,80): M2=-1: A2$="1": IF A1$="D10"OR A1$="D20"OR A1$="D30"THEN 6
  15 GOTO 620
615 GOSUB 5040: ON T+1GOTO 565,565,565: A$="360 KB": IF VAL(STR(E$,6,3),3)=4160T
  HEN A$="1.2 mB": PRINT AT(5,0,80);HEX(0E);STR(A$,,LEN(A$));" Floppy Drive. Plea
  se Select": PRINT " [1] CS/2200 format": PRINT " [2] DOS format": LINPUT " "--
  A2$: PRINT HEX(06);
617 IF A2$="2"THEN 620: IF A2$="1"THEN 620: GOTO 615
620 PRINT AT(9,0);HEX(0E);"Mount disk to be formatted and press RETURN:": REM -
  ^ROMPT
625 GOSUB '201(M$): REM - DISPLAY MESSAGE (IF ANY)
  
```



```

530 PRINT AT (22,52); "RETURN - Next Field"; AT (23,52); "FN/TAB - Previous Screen"; HEX(
01)
650 F=1; KEYIN K$: IF K#=HEX(7E)OR K#=HEX(7F)THEN 1B80: IF K#<>HEX(0D)THEN 650
700 REM %GET & DISPLAY INDEX INFORMATION (IF INDEX SECTORS > 0)
710 IF M2=-1THEN SELECT #1(A1$>: ERROR GOSUB '255: GOTO 610: REM - SELECT DISK h
hh
725 V$(1)="24": V$(2)=" ": V$(3)="0": REM - INITIALIZE THE INDEX INFORMATION FIE
L: X=0
730 $OPEN 732,#1: ERROR GOSUB '255: GOTO 610: REM - "HOG" THE DISK
731 GOTO 740: REM - SKIP THE "CAN'T $OPEN" MESSAGE
732 M$="Drive Already In Use or Not Ready": PRINT HEX(07): GOTO 610: REM - DISPL
AY MESSAGE IF CAN'T $OPEN DRIVE
740 IF D0$="340"THEN 760: G$(11)=HEX(FF): $BIOGETDISKTYPE#1(0200030F122206000700
70A04000870B,G$(11)): REM - GET THE DRIVE TYPE (DPU TYPE)
750 IF STR(G$(1),6,3)=HEX(00 00 00)THEN 752: M$="Drive Not Responding": PRINT HEX
(07): GOTO 610
752 IF G$(11)=HEX(D0)THEN 760: IF G$(11)=HEX(C0)THEN 760: IF G$(11)=HEX(FF)THEN
PRINT AT(6,0,80);"No disk controller at address": GOTO 565
760 DATA LOAD BA T#1,(0)S$(1): ERROR GOSUB '255: IF M<>93THEN 610: M$=" ": GOTO 8
00: REM - GET FIRST INDEX SECTOR. FORMAT DISK IF FORMAT ERROR. REPORT ALL OTHE
R ERRORS
761 IF STR(S$(0),1,1)<>HEX(02)THEN 765: IF STR(S$(0),2,2)=HEX(00 00)THEN 800: REM
- IF 3 BYTE INDEX = 00000 THEN FORMAT IT
762 CONVERT VAL(STR(S$(0),2,2),2)TO V$(1),(#####): CONVERT VAL(STR(S$(0),4,3),3)TO
V$(2),(#####): CONVERT VAL(STR(S$(0),7,3),3)TO V$(3),(#####): REM - CHANGE
THE 3-BYTE VALUES TO STRINGS
763 PRINT AT(14,29);"Index Type = 3-Byte": GOTO 775
765 IF STR(S$(0),2,1)=HEX(00)THEN 800: REM - IF INDEX SECTORS = 00000, FORMAT IT
770 FOR I=1TO 3: IF I=1THEN CONVERT VAL(STR(S$(I),2,1)TO V$(1),(#####): ELSE CO
NVERT VAL(S$(I),2)-1TO V$(I),(#####): IF V$(I)="00000"THEN V$(I)="0": ELSE V$(I)
=STR(V$(I),POS(V$(I)<>"0")): NEXT I: REM - CHANGE THE 2-BYTE VALUES TO STRINGS
775 PRINT AT(13,27);"Index Sectors = ";V$(1): AT(15,29);"Index Sectors = ";V$(
1): AT(16,29);"End Cat. Area = ";V$(3): AT(17,29);"Current End = ";V$(2): REM -
5. GIVE HIM THE CURRENT INDEX
776 IF STR(S$(0),1)=HEX(01)THEN PRINT AT(15,51);"1": IF STR(S$(0),1)=HEX(02)THEN
PRINT AT(15,51);"2"
780 PRINT HEX(07): GOSUB '201("Are you sure (Y/N)?"): M$=" ": GOSUB '204: IF K#=
"N"THEN 510: REM - GIVE HIM THE FINAL WARNING
795 REM %TRASH THE SUCKER!
800 M$=HEX(02 04 04 00 0E)&"(Formatting)"&HEX(02 04 00 00 0E): GOSUB '201(M$): R
EM - TELL HIM THE NASTY THINGS WE'RE DOING
810 IF A2$="2"THEN 812: $FORMATDISK T#1: ERROR GOSUB '255: GOTO 610: REM - FINAL
LY! FORMAT THE DISK
811 GOTO 820
812 REM ! PC FORMAT: SELECT #1(A1$>: G$=ALL(20): $BIO#1(0600070070A0400288D07040
01306A10680240018B67,G$)
813 IF STR(G$,6,3)=HEX(00 00 00)THEN 820
814 IF STR(G$,6,3)<>HEX(00 00 00)THEN D0: PRINT HEX(07);: M$=HEX(02 04 02 0E)&"D
isk Error - Press any Key "&HEX(0F): GOSUB '201(M$): KEYIN K$: M$=ALL(" "): ENDD
0: GOTO 610
815 GOTO 510
820 GOSUB '201("Format Completed"): REM - WE MADE IT
822 F=2
825 REM %GET NEW INDEX PARAMETERS & SCRATCH THE DISK
827 GOSUB '300: REM - CHECK DISK TYPE
830 PRINT AT(6,0,80);HEX(0E);"Enter new index information and press RUN:": REM -
CHANGE THE PROMPT
835 IF STR(S$(0),1)=HEX(01)THEN V$(2)="NEW": ELSE V$(2)="OLD": REM - INITIALIZE
THE STRUCTURE FIELD
840 PRINT AT(13,0,400);HEX(02 04 02 04 0F); AT(13,29);"Index Sectors = ";HEX(0E);STR(V$(1));HEX(0F); AT(16,29);"End Cat. Area =
";HEX(0E);STR(V$(3));HEX(0F); AT(17,29);"Structure = ";HEX(0E);STR(V$(2),3);
HEX(02 04 02 00 0F);" (OLD or NEW)": REM - DISPLAY THE FIELDS
841 IF STR(J$,9,1)<>"T"OR STR(S$,3)="0"THEN 845: IF T<>15AND Z<"3F"THEN 845: P
RINT AT(17,64);" or 3 (key TRI)": REM - IF TURBO ALLOW 3 BYTE OPTION
845 PRINT AT(21,52);"RETURN - Next Field"; AT(22,52);"RUN - Accept parameters"

```



```
REM - ADDITIONAL DEFINITIONS
850 KO=1: REM - RESET FIELD COUNTER
855 ON KOGOSUB 875,880,885: REM - PROCESS THE CURRENT FIELD
858 GOSUB '201(" "): REM - ERASE THE MESSAGE (IF ANY)
860 ON POS(HEX(82 7E 7F)=K#)GOTO 900,520,520: REM - RUN , FN / TAB
865 KO=KO+1: IF KO>3THEN KO=1: GOTO 855: REM - RETURN PRESSED, ADVANCE TO NEXT F
FIELD
870 GOSUB '202(15,45,V$(1),5,"#",HEX(7F 7E)): V$(1)=I0$(1): RETURN: REM - ACCEPT
INDEX SECTORS
880 GOSUB '202(16,45,V$(3),8,"#",HEX(7F 7E)): V$(3)=I0$(1): RETURN: REM - ACCEPT
END OF CATALOGED AREA
885 GOSUB '202(17,45,V$(2),3,"A",HEX(7F 7E)): V$(2)=I0$(1): $TRAN(V$(2),"AaBbCcD
dEeFfGgHhIiJjKkLlMmNnOoPpQqRrSsTtUuVvWwXxYyZz")R
886 IF V$(2)="OLD"OR V$(2)="NEW"THEN RETURN: IF T<>-15AND Z#<"3F"THEN 887: IF V$
(2)="TRI"AND STR(J#,9,1)="T"AND STR(S#,3)>"0"THEN RETURN
887 IF T=-15AND STR(J#,9,1)="T"AND STR(S#,3)>"0"THEN 888: GOSUB '201(
"Index Structure Must be 'OLD' or 'NEW'"): PRINT HEX(07): GOTO 885: REM - ACCEPT
INDEX STRUCTURE
888 GOSUB '201("Index Structure Must be 'OLD', 'NEW' or 'TRI' for 3 Byte"): PRIN
T HEX(07): GOTO 885: REM - ACCEPT INDEX STRUCTURE FOR TURBO
900 GOSUB '201("Creating Disk Index"): REM - TELL HIM WHAT WE'RE UP TO
910 CONVERT V$(1)TO V1: ERROR PRINT HEX(07): GOSUB '201("Index Sectors may not b
e blank"): GOTO 850: REM - CHANGE INDEX SECTORS INTO A NUMERIC VALUE
911 IF V$(2)="TRI"AND STR(J#,9,1)="T"AND STR(S#,3)>"0"AND Z#>"3F"THEN 913: REM
- 913 IF 3 BYTE, TURBO, NON=0 ADDR, & DS 3F PROM MIN
912 IF V$(2)="TRI"AND STR(J#,9,1)="T"AND STR(S#,3)>"0"AND T=-15THEN 913: IF V1>=
1AND V1<=255THEN 915: PRINT HEX(07): GOSUB '201("Index Sectors must be from 1 to
255"): GOTO 850: REM - VALIDATE INDEX SECTORS VALUE
913 IF V1>=1AND V1<=65535THEN 915: PRINT HEX(07): GOSUB '201("Index Sectors must
be from 1 to 65535 for 3 Byte"): GOTO 850: REM - VALIDATE INDEX SECTORS VALUE F
OR 3 BYTE
915 CONVERT V$(3)TO V3: ERROR PRINT HEX(07): GOSUB '201("End Cat. Area may not b
e blank"): GOTO 850: REM - CHANGE END OF CATALOGED AREA INTO A NUMERIC VALUE
918 IF V1<=V3THEN 918: PRINT HEX(07): GOSUB '201("End Cat. Area less than Index
Sectors"): GOTO 850
918 IF V$(2)="TRI"AND STR(J#,9,1)="T"AND STR(S#,3)>"0"AND T=-15THEN 920: IF V$(
2)="TRI"AND STR(J#,9,1)="T"AND STR(S#,3)>"0"AND Z#>"3F"THEN 920
919 IF V3>=V1AND V3<=65535THEN 920: PRINT HEX(07): GOSUB '201("End Cat. Area mus
t be less than 65535"): GOTO 850: REM - VALIDATE 2 BYTE END CAT. AREA
920 IF V$(2)="OLD"THEN SCRATCH DISK T#1,LS=V1,END=V3: ERROR GOSUB '255: GOSUB '2
01(M#): GOTO 850: REM - BUILD THE OLD INDEX STRUCTURE & HANDLE ERROR
930 IF V$(2)="NEW"THEN SCRATCH DISK T#1,LS=V1,END=V3: ERROR GOSUB '255: GOSUB '
201(M#): GOTO 850: REM - BUILD THE NEW INDEX STRUCTURE & HANDLE ERROR
933 IF STR(S#,3)="0"THEN 940: IF T<>-15AND Z#<"3F"THEN 940: IF V$(2)<>"TRI"AND S
TR(J#,9,1)<>"T"THEN 940
934 IF V1<256THEN 935: GOSUB '201("Be patient, large indices take time")
935 SELECT 3ON: SCRATCH DISK T#1,LS=V1,END=V3: ERROR GOSUB '255: GOSUB '201(M#)
: GOTO 850: REM - BUILD THE 3 BYTE INDEX STRUCTURE & HANDLE ERROR
940 GOSUB '201("Index Created - Press Any Key to Exit"): PRINT HEX(07): KEYIN K#
: REM - LET HIM KNOW WE'RE FINISHED
999 $CLOSE #1: GOTO 520: REM - FORMAT/SCRATCH COMPLETED. "UNH06" THE DISK & GO
TO THE DISK ADDRESS MENU
1870 DEFFN'126
1880 DEFFN'127: GOSUB '201("FN/TAB keyed")
1890 IF F=1THEN 510
2000 COM CLEAR: GOSUB '201("Returning to System menu"): $PSTAT=" ": LOAD DC I'
@MENU": END: REM - THAT'S ALL SHE WROTE
5025 DEFFN'100(S#): IF S#="340"THEN 5035: $TRAN(S#,"AaBbCcDdEeFf")R: IF POS("DB0
")>0AND POS("1235&7"=STR(S#,2))>0AND VER(STR(S#,3),"P")>0THEN 5030: Q#="I": F
ETURN
5030 IF POS("35"=S#)=0OR POS("123"=STR(S#,2))=0OR STR(S#,3)>"0"THEN 5035: IF S1
R(S#,1)="3"THEN STR(S#,3)="1": STR(S#,1)="D"
5035 Q#=" ": RETURN
5040 REM %1.Get disk status
5050 A$=ALL(20): STR(B#,2,1)=STR(B#,2,1)AND HEX(33): SELECT #2<B#
5080 $BIT0BETDISKTYPE#2(0200030F12220600070070A04000870B.G#0)
```

```

5100 DN T60SUB 5260,5250,5240,5230: PRINT K1#
5110 D#:=D#(1)
5130 RETURN
5140 PRINT "Unknown disk type check connector": D#(1)=" ": RETURN
5150 REM %T, Check for DS cabinet
5160 G#:=HEX(30): STR(G#,2,7)=ALL(00): %GI05STATUSREQUBS7#2(0E1406012E206900710
002B8D07040&A1068164000B7051A00C340,6#)B#: STR(E#,VOL-STR(G#,3,1)2)
E O T=VAL(E#)-47
5180 G1#:=STR(G#,8,1)AND HEX(10): IF G1#:=HEX(00) THEN 5190: K1#="Disk unavail
": RETURN
5190 IF STR(G#,8,1)>HEX(00) THEN 5200: IF STR(E#,1,1)=
ND STR(E#,1,2)=HEX(45 33) THEN 5210
5200 REM "specified disk is not a DS". RETURN
5210 STR(E#,1,2)=HEX(33 45)
5220 RETURN
5230 K1#="DS cabinet": RETURN
5240 K1#="2275 type": RETURN
5250 K1#="Phoenix type": RETUR
5260 K1#="2270 type": RETURN
6000 REM % GET DISK STATUS
7000 DEFFN 201(0#): REM % DISPLAY M# IN LOWER LEFT CORNER OF THE DISPLAY
7010 PRINT AT(23,0,50):HEX(05):%R#(0#):HEX(02): KEY-- DISPLAY MESSAGE
7020 RETURN
7030 DEFFN 202(11,12,10#(1),13,10#(14)): REM % ACCEPT FIELD (11,12)50W,50
)=DATA 13=LEN 10#=#VE# MASK 10#=#EXIT SFKEY5)
7040 MAT REDIM 10*(1)13: REM - CHANGE LENGTH OF FIELD TO LENGTH SPECIFIED
7050 PRINT HEX(02 04 02 04 0E): REM - SET ATTRIBUTES TO BRIGHT/UNDERSCORE
7060 I0=1: REM - INITIALIZE CURSOR POSITION
7070 PRINT HEX(06):AT(11,12):STR(10*(1)): REM - DISPLAY THE FIELD
7080 PRINT AT(11,12+I0-1):HEX(02 05 0F):: REM - POSITION CURSOR
7090 KEYIN K#:7200: REM - WAIT FOR A KEY. BRANCH IF SF KEY PRESSED
7100 PRINT HEX(06):: REM - "NORMAL" KEY PRESSED: SHUT OFF THE CURSOR
7110 ON POS(HEX(20 0B 0D 82)=K#)GOTO 7130,7150,7180,7190: REM - BRANCH IF SPACE,
BACKSPACE, RETURN OR RUN/EXEC
7120 IF VER(K#,10#)<>1 THEN 7080: REM - IF NOT A VALID KEY, WAIT FOR ANOTHER
7130 IF I0#:=HEX(00) THEN 7080: STR(I0*(1),I0,1)=K#: PRINT K#: REM - IF ACCEPTING
CHARACTERS, PUT THE CHARACTER INTO THE FIELD & PRINT IT
7140 IF I0<13 THEN I0=I0+1: GOTO 7080: REM - INCREMENT CURSOR POSITION & UPDATE I
F
7150 IF I0=1 THEN 7080: REM - BACKSPACE PRESSED: GET ANOTHER KEY IF AT LEFTMOST C
OLUMN
7160 IF I0<13 OR STR(I0*(1),I0,1)=" " THEN I0=I0-1: REM - DON'T DECREMENT CURSOR P
OSITION IF AT RIGHTMOST POSITION & IT'S NON-BLANK
7170 STR(I0*(1),I0,1)=" ": GOTO 7070: REM - REMOVE THE CHARACTER & UPDATE THE CR
T
7180 K#:=HEX(20): GOTO 7280: REM - RETURN PRESSED: TRANSLATE 11 TO HEX(20) & EXIT
7190 GOTO 7280: REM - RUN/EXEC PRESSED: EXIT
7200 PRINT HEX(06):: REM - SF KEY PRESSED: SHUT OFF THE CURSOR
7210 IF POS(11#:=K#):NOTHEN 7280: REM - IF AN EXIT KEY, EXIT
7220 IF I0#>HEX(00) THEN ON POS(HEX(4B 49 4A 4C 4D)=K#)GOTO 7230,7240,7250,7260,
7270: GOTO 7080: REM - IF MODIFIABLE, BRANCH TO ERASE, DELETE, INSERT, RIGHT OR
LEFT
7230 STR(I0*(1),I0)=" ": GOTO 7070: REM - ERASE PRESSED: DO IT & UPDATE THE CRT
7240 IF I0<13 THEN STR(I0*(1),I0)=STR(I0*(1),I0+1): STR(I0*(1),I3,1)=" ": GOTO 70
70: REM - DELETE PRESSED: DO IT & UPDATE THE CRT
7250 IF I0<13 THEN MAT COPY -STR(I0*(1),I0,I3-I0) TO -STR(I0*(1),I0+1): STR(I0*(1)
,I0,1)=" ": GOTO 7070: REM - INSERT PRESSED: DO IT & UPDATE THE CRT
7260 IF I0<13 THEN I0=I0+1: GOTO 7070: REM - RIGHT PRESSED: MOVE CURSOR & REPOSIT
ION IT
7270 IF I0>1 THEN I0=I0-1: GOTO 7070: REM - LEFT PRESSED: MOVE CURSOR & REPOSITIO
N IT
7280 PRINT HEX(02 04 02 00 0F): REM - SET THE ATTRIBUTES TO BRIGHT (NORMAL)
7290 MAT REDIM 10*(1)180: REM - RESET THE FIELD LENGTH TO ITS MAXIMUM
7300 RETURN
7310 DEFFN 204: REM % RETURN K# = "Y" OR "N"

```



```

7325 IF K#=HEX(7E)OR K#=HEX(7F)THEN K#="N"
7330 $TRAN(K#,"Yynn")Rr: IF K#<>"Y"AND K#<>"N"THEN
7340 RETURN
8000 DEFFN'300: REM %1.Get disk status
8020 B#=D0#: A#=ALL(20): STR(B#,2,1)=STR(A#,2,1)AND HEX(33): SELECT #2<B#>
8030 #6I0GETDISKTYPE#2(0200030F12220600070070A04000870B,B#(1))
8040 T#=B#(11): T=POS(HEX(C0 D0)=T#): IF TK>2THEN RETURN
8050 REM %1.Check for DS cabinet
8060 G#=HEX(30): STR(G#,2,7)=ALL(00): #6I0STATUSREQUEST#2(0E140F0012E20600070070
A040028BD070406A106816400087051A00C340,B#)G#:STR(X#,VAL(STR(G#,5,1)))
8070 T=VAL(X#)-47: IF TK>4THEN RETURN: REM - RETURN IF NOT A DS
8090 IF STR(X#,2,1)=HEX(00)THEN STR(E#,18,1)=HEX(00): G1#=STR(G#,8,1)AND HEX(10)
: IF G1#>HEX(00)THEN 8120
8100 IF STR(G#,8,1)<>HEX(00)THEN 8130: X#(1)=X#
8110 Z#=STR(X#(1),4,2): REM - Z#=DS FROM REVISION
8115 RETURN
8120 GOSUB '255("Disk unavailable")
8125 RETURN
8130 GOSUB '255("Specified disk is not a DS")
8135 RETURN
9000 DEFFN'255: REM % ERROR HANDLER (SETS M#=ERROR DESCRIPTION & REEPS)
9005 M=ERR: REM - GET THE ERROR NUMBER
9010 M#="" : REM - INITIALIZE MESSAGE
9015 IF MK90THEN 9025: RESTORE LINE9020,M-89: READ M#: REM - GET MESSAGE IF DISK
ERROR
9020 DATA "Disk Hardware Error","Disk Drive Not Ready","Disk Drive Time-Out","Di
sk Format Error","Disk Format Key Engaged": REM - DISK ERRORS
9021 DATA "Disk Seek Error or Platter Protected","Disk CRC Error","Disk LRC Erro
r","Bad Sector Address/Platter Not Mounted","Verify Error": REM - DISK ERRORS (C
ONT.)
9025 IF M=48THEN M#="Illegal Device Specification"
9030 IF M#<>" "THEN 9040: M#="Unexpected Error ### Occurred": CONVERT MTO STR(M#
,3),(###): REM - CATCH-ALL
9040 PRINT AT(23,0,33);HEX(0E);STR(M#);HEX(01): REM - DISPLAY MESSAGE
9050 PRINT HEX(07): RETURN

```

# FORMAT VER 2.00.00

5/18/93 FAILS ON VLSI O/S w/ ERROR P34 (ILLEGAL VALUE) ON LINE 827

827 ~~GOSUB~~ 300:REM  
↑

FIX

827 GOSUB 250:REM - CHECK DISK TYPE WAS 300

8000 DEFFN 250:REM%↑. GET DISK STATUS WAS 300

LINE 935. SELECT 3 ON CAUSES S19 ERROR. ON VLSI + 386 1.1z

SCRATCHDISK ST#1, LS=V1, END=V3 CAUSES S16 ERROR ON VLSI + 386 1.1z

CONVENTION:

VLSI/TURBO PLACE REM%% IN FRONT OF SELECT 3 ON + SCRATCHDISK  
386 REM

5/24/93 IF CREATE A 3 BYTE ADDRESS ON A TURBO + THEN REFORMAT + SCRATCH w/ AN "OLD" OR "NEW" INDEX, REMAINS AS A 3 BYTE.

FIX: CHANGE

FROM: 933 IF STR(S#,3) = "0" THEN 940: IF T <> -15 AND Z# < "3F" THEN 940: IF V#(2) <> "TRI" AND STR(J#,9,1) <> "T" THEN 940

TO 933 : IF V#(2) <> "TRI" OR STR(J#,9,1) <> "T" THEN 940

WHEN READING A 3 BYTE INDEX ON A DISK TO BE FORMATTED, THE "END CAT. AREA" + "CURRENT END" FIELDS ARE 1 LARGER THAN THEY SHOULD BE.

FIX: ADD THE FOLLOWING CHANGE:

762 CONVERT VAL(STR(S#( ), 2, 2), 2) TO V#(1), (#####)

: CONVERT VAL(STR(S#( ), 4, 3), 3) - 1 TO V#(2), (#####)

: CONVERT VAL(STR(S#( ), 7, 3), 3) - 1 TO V#(3), (#####)

: REM - CHANGE THE 3-BYTE VALUES TO STRINGS



6/11/93 CANNOT FORMAT PHOENIX REMOVABLE, ASKS YOU TO RE-ENTER ADDRESS.  
FIX 615 ON T+1 GOTO 565,565,620 WAS 565

- ADDED MESSAGE AS FOLLOWS IF TRIED TO FORMAT 2270.  
5260 K1#="2270 type not supported":RETURN

- ADDRESS B10 NOT RECOGNIZED AS FLOPPY ADDRESS AND FAILS TO CHECK TYPE.  
FIX 5030 IF POS..... THEN 5035: IF STR(SA,,1)="3" THEN STR(SA,3)  
="1": STR(SA,,1)="D": A1# = SA

- DOES NOT RECOGNIZE LVP FLOPPY & WILL NOT ALLOW FORMAT. ~~RE-ENTER ADDRESS~~  
~~RE-ENTER ADDRESS~~ SHOWS MESSAGE "UNKNOWN DISK TYPE, CHECK CONNECTOR".  
FIX

5090 T# = G\$(11): T = POS(HEX(CO DO) = T#): IF T# = HEX(20) THEN T = 5:

IF T = 0 THEN 5140: IF T = 2 THEN GOSUB 5150

5100 ON T GOSUB 5260,5250,5240,5230,5270: PRINT K1#

5270 K1# = "LVP Floppy": RETURN

615 GOSUB 5040: ON T+1 GOTO 565,565,620: IF T = 5 THEN 620: A# = "360 KB":

IF VAL(STR(E#,6,3),3) = 4160 THEN A# = "1.2 MB"

617 PRINT AT(5,0,80); HEX(OE); STR(A#, LEN(A#)); "Floppy Drive. Please

Select": PRINT " [1] CS/2200 Format": PRINT " [2] DOS Format":

LINPUT " " - A2# : PRINT HEX(O6);

Mount message overwrites drive type (shown only w/ removable address)

FIX 620 PRINT AT(10,0); HEX(OE); "Mount disk to be formatted and press

RETURN:";: REM - PROMPT

8/4/93 LVP DSDD COMES UP AS PHOENIX TYPE.

5100 IF T = 2 AND STR(G#,5,1) = HEX(10) THEN T = 5: ON T GOSUB 5260,5250,5240,5230,  
5270: PRINT K1#

1/20/94 ERROR S19 ON LINE 935 ON CS/386. DOES NOT RECOGNIZE SELECT 3 ON FOLLOWING  
REM%%. ERROR S16 ON LINE 935 ON CS/386. DOES NOT RECOGNIZE SCRATCHDISKS  
FOLLOWING 2 REM%%.

FIX: REMOVE 2<sup>ND</sup> % FROM BOTH REM COMMANDS ON LINE 935

ADDED REM ON LINE 92 TO REMOVE REM%<sub>s</sub> ON LINE 935 FOR 3 BYTE SCRATCH ON TURBO.

# C GENPART

11/6/92

IF CURRENT PARTITION > 8.M. UNABLE TO EXECUTE W/ SF'15. GET MESSAGE "PARTITION TOO LARGE".

ON LINE 610 CHANGE: IF X > A(1) TO IF X > S9

3/2/93 ALWAYS GET "INVALID CONFIGURATION" ON VLSI SYSTEM WHEN BOOTING FROM A CONFIGURATION FOR THE 2<sup>ND</sup> TIME. CREATE ALWAYS WORKS. PROBLEM CAUSED BECAUSE BYTE 22 IN THE 1<sup>ST</sup> SECTOR OF C-SYSFILE FOR CURRENT = 00<sub>HEX</sub>. MUST BE 20<sub>HEX</sub> FOR VLSI.

FIX: ADDED LINE 11 DIM JB(1): JB = STR(C#9,1)

GET CPU TYPE

2760 GOSUB'191: IF O1 < 4 AND FØ <> -2 THEN U# = " " : IF JB = "M"

THEN 2770

2765 MIB = HEX(00 40 80 C0): FOR X = 1 TO 4: GOSUB'193(X): NEXT X:

GOTO 2775

2770 MIB = HEX(20 40 80 C0): FOR X = 1 TO 4: GOSUB'193(X): NEXT X

2775 RETURN

ADD NEW  
BLANK WAS ALL  
ON 2760

3/3/93 IF LOADED TURBO CONFIG W/ > 16 PARTITIONS ON 386 WOULD BLOW O/S.

400 IF O1 = 4 AND W > 16 THEN FØ = -2: IF ~~JB~~ JB = "W" AND W > 16 THEN 403:

IF FØ > -1 THEN 405

403 MIB = "INVALID CONFIGURATION FOR THIS CPU.": GOSUB'196: GOTO 1150

WAS ON  
LINE 460

4/28/95 MYP SYSTEM WITH 64K MEMORY WILL NOT SHOW SF'15 ON BOOT

FIX WILL NOT LET YOU BOOT. GET MESSAGE "SYSTEM HAS ALREADY BEEN CONFIGURED."

MOMENTARILY. ON LINE 22 NEED TO CHANGE SPACEK TEST TO < 61 INSTEAD OF < 62.

FIX 22 LI = Ø: IF #PART = 1 AND #TERM = 1 THEN LI = 1: IF CØB = "M" AND SPACEK < 61 THEN LI = Ø:

IF CØB = "W" AND SPACEK < 1024 THEN LI = 0: IF LI = Ø THEN 24: LI = Ø: M# = \$PSTAT(2): ERROR LI = 1



8/28/95 IF LOAD A CONFIGURATION WITH MORE THAN 16 PARTITIONS ON CS TYPE CPU, GETS HUNG IN ENDLESS LOOP BETWEEN LINES 2120 & 2050. EDITED LINE 2120 TO PRINT OUT MESSAGE & RETURN TO LOAD CONFIGURATION SCREEN.

~~FIX 2120 IF U <> INT(U) OR U <= 0 OR U > UI THEN MI\$ = "INVALID NUMBER OF PARTITIONS FOR THIS CPU." : GOSUB 196 : GOTO 1150~~

FIX 2120 IF U <> INT(U) OR U <= 0 OR U > UI THEN 2122 : ELSE GOTO 2125

2122 MI\$ = "INVALID NUMBER OF PARTITIONS." : GOSUB 196 : ~~FIX~~ GOTO 1150

11/14/96 IF SET GENPART FOR AUTO EXECUTION BY PLACING REM IN FRONT OF GOTO 1150 ON LINE 100, STILL PROMPTS FOR "RECONFIGURATION PASSWORD?"

FIX ~~103 GOTO 425~~

103 REM GOTO 425 : REM % ACTIVATE GOTO TO ELIMINATE PROMPT FOR PASSWORD ON AUTO EXECUTION



LATEST = GENPART

INVALID CONFIG ON VLSI

OLD GENPART

FO = 1 THEN OK

RESET

FO = -2

LINE 400 HAS INVALID CONFIG MESSAGE

FO	10	95	130	176	204	340	345	352	380	400	590	1210	1230		
												1232	2010	2760	3090

OLD GENPART

LINE 1232 DEFINES FO

IF U# = " " THEN FO = 1      U# = 20 so FO = 1

IF C# = "W" AND U# = " " THEN FO = -FO      C# = M HD      386 = -1

IF C# <> "W" AND U# < " " THEN FO = -FO      TURBO?

NEW ~~FORMAT~~ GENPART

1220 U# = W#

U# = HEX 20

W# = HEX 00

13,1220,1370,1372,1374,1375  
2735

1232 IF U# < " " THEN FO = 2

SETS FO = 2 BECAUSE U# = 00<sub>8</sub>

IF U# = " " THEN FO = 1

VLSI 1

IF C# = "W" AND U# = " " THEN FO = -FO

386 -1

IF C# <> "W" AND U# < " " THEN FO = -FO

W# - FIRST BYTE CHANGE TO 00

13,1220,1360,2730,2735

1300 DATASAVE DC #0, F2#, W

ESYSFILE

OLD

CONFIG FILE

20, 56, 5010, 5050

20 FO# = 'ESYSFILE'

16 FO# = 'ESYSFILE'

NEW

16, 56, 285, 5010, 5050

FO#

DIM

ESYSFILE

10

16

58

870

940

950

970

980

1210

1340

2310

2320

DATALOAD DC OPEN T#FO, FO#:

LIMITS

DATASAVE FC OPEN

DATASAVE DC OPEN

DATALOAD DC OPEN

DATALOAD DC OPEN

DATALOAD DC OPEN

DATALOAD DC OPEN

PSTAT  
 FORMAT  
 MOVEFIL  
 BACKUP  
 RECOVER  
 INSTALL  
 REF  
 DRIFU  
 COMRTIAN

OLD GENPART  
 LIST V U#

12 72 95 905 1220 1232 1375 1970 2160 2180 2710 2760 2800  
 3885

GENPART

CHANGE LINE 2760

MIB# = HEX(00 40 80 C0)  
 TO " 20 "

C# = #PSTAT(1) 10<sup>dim</sup> 16 C# = STR(C# 9)  
 M# 2 135 215 890 905 1220 1375 1740 2750 2780  
 X# 2

~~009~~

J#

11 DIM J#(1): J# = STR(C# 9, 1)  
 2760 GOSUB '191: IF 01 < 4 AND # < > -2 THEN U# = " ": IF J# = "M" THEN 2770  
 2765 MIB# = HEX(00 40 80 C0): FOR X = 1 TO 4: GOSUB '193(X): NEXT X: GOTO 2775  
 2770 MIB# = HEX(20 40 80 C0): FOR X = 1 TO 4: GOSUB '193(X): NEXT X ~~RETURN~~  
 2775 RETURN

Z730 DATALOAD DC #0, F2#, W, W9#, W0#, W1#, W2#, W3#( ), W4#( ), W5#( )

F2# = CURRENT

THIS SETS THE 1ST BYTE TO 00 CAUSING PRBS ON VLS1

OLD FORMAT 12, 1220, 2730, 2740

### SYSFILE

CHANGED BYTE 22 IN @SYSFILE FROM 00 TO 20 & WORKS 1ST TIME THROUGH BUT WABN SAVE CONFIG<sup>SF15</sup> FAILS AGAIN BY CHANGING IT BACK TO 00.

AT INITIAL CONFIG SCREEN U# = 20  
W# = 20  
W9# = 2020

SF 15 U# = 20 W# = 20 W9# = 20

OK TO EXEC Y U# = 20 W# = 20 W9# = 20

RECONFIG PASSWORD U# = 00 W# = 20 W9# = 20

LIST V U# 13 72 95 905 1220 1232 1375 1970 2125 2160  
2180 2710 2760 2780 2790 2800 3085 9990

Z720 DATALOAD "@SYSFILE"

Z780 U# = U# AND HEX(1F) XOR STR(MI#, X)

HEXPRINT MI# 004080C0 2020 X=1  
MI# 12 395 400 630 640 680 1210 1340 1440 1445 2310  
2760 2780 4120 4125? 4140 4200 4240 4345 9110  
MI# = (00 40 80 C0)

## NEW DISK

~~SCSI BACKUP NO~~

~~CBACKUP NO~~

~~SCSI RESTORE NO~~

~~CRECOVER NO~~

~~SCSI CONFIG NO~~

## FAILING CBOOT

9010 DATA "CBOOT", " \_\_\_\_\_ ", 6, 0

9080 "CDG2", "TURBO", "A"  
CDG1 M

## VLSI

DIAGS COME UP NOT A PROGRAM RECORD

CHANGE M TO P. ON LINE 9035 & GET D82 FILE NOT FOUND

IF BOOT FROM NON-VLSI CONFIG FAILS, MUST RESET & RERUN; SHOULD REJECT

1360 PRINT OK CONFIG #PART CPU

1374 IF 01 > 3 AND W# < " " THEN Y# = "y" > IF 01 < 4 AND W# = " " THEN Y# = "y"

1375

386 SAYS Y FOR TURBO CONFIG

INVALID CONFIG FOR THIS CPU IF USE VLSI CONFIG

TURBO CONFIG w/ 40 PART ACCEPTED BUT BLEW CPU

DIAGS OK

CREATEB CONFIG OK

SAVE OK



W = # TERMINALS

400

386 LOAD VLS1

SF15 HALT

395 2000 DEFFN' 204 E# = 20 U = 5 K = 1

1730 DEFFN' 206 I = 1 P#1

1740

1750 B# 2020 -

1760 J = 1 U = 5

1770 F# = -1 B = 16 R(J) = 56

1780 I = 1 M = 2 B16 S(J) = 0 X = 0

1785 Z = 0

1790 X# = 208 B#(J) = Y

1810 E# = " "

2000 E# = " "

395 IF CDB = "W" THEN GOSUB 2710

2710 U# = " " PRINT CPU # IF

395 LI = 1

400 OI = 5 W = 5 FO = -1 MI# = INVALID CONFIG

PAK?

386

PRINT# ?

M# PAK 572.5

386

2710 OR 386

PAK?

PRINT# ?

~~2020~~ ~~A(I) = AVAILABLE MEM~~  
SA = TOTAL SYSTEM MEM

386 SYS LOAD TURBO

SF'15 HALT 395

2000 GOSUB' 204 E# = 4C : L" U = <sup>PAR 40</sup> K = 0

~~2010 E# = "~~

~~3010 DEFFN' 201 K = 0~~

~~3000~~ 1730 DEFFN' 206 I = 1 P# = 1

1740 S(J) = 62.5 <sup>PAR 5126</sup> J = 1 L# = 2 62.5 1 Y T# = 1 - 40

C# = YYY - P# = SYSTEM

1750 B# = 202020 -

1760

1770 FO = 2 B = 1 R(J) = 5692

1780 I = 1 M = 2 B = 1 ~~S(J)~~ S(J) 62.5 X = -34823.2

1810 E# = " "

2000 E# = " "

395

2710 U# = 05 <sup>CPU #</sup> U9 = CPU #

395 LI = 1

400 OI = 5 <sup>PAR 5126</sup> W = 40 FO = 2

405 X# = " "

1860 X# = " F"

1870 J = 0

1900

1920 R(J) = 8192 <sup>REMAINING MEM</sup> S(I) = 62.5 <sup>TOT MEM</sup> <sup>PAR 5126</sup>

R(J) KEEPS TRACK OF MEMORY LEFT

1950

1900

405

415

400 IF OI=4 AND W>16 THEN F0=-2: IF C0#="W" AND W>16 THEN 403: IF F0>-1  
THEN 405

403 MI#="INVALID CONFIG"

386 sys LOAD 386X CONFIG

SF'15 HALT 395

2000 DEFFN' 204 E# = 20 U = 8 K = 1

1730 DEFFN' 206 I = 1 P# = 1

1740 S(J) = 100 J LA = 2 TB = 01 02 03 04 05<sub>8</sub>

C# = Y Y Y - PH = SYSTEM

1750 B# = 202020 -

1760

1770 F# = 2 B = 1 R(J) = 7392

1780 I = 1 M = 2 B = 1 S(J) = 100 X = -97769.6

1810

2000

395

2710 U# = CPU # U# = 02

395 L1 = 1

400 OI = 5 W = 8 FO = 2

405

1860 X# = "E"

1870

1900

1920 B#(1) = N

1930 S(1) = 100 SUBTRACTS PART SIZES FROM TOT MEM

1950

1900

405

415



VS OFFICE

Monday

03/09/92 08:28 am

To: Michael  
From: Kirit  
Subject: Print Driver bug

W0000600 6FLT3  
Security: Limited  
Date Received: 03/09/92

-----  
Mike

Change line 5120 of @GENPART as follows:

```
5120 IF STR(Q2$(I),2,2)=HEX(00)THEN STR(Q2$(I),2,2)=HEX(30 30)
      : HEXPACK STR(T2$(I),9,1)FROMSTR(Q2$(I),2,2)
      :IF Q3$(I)=" "OR Q3$(I)=HEX(00 00)THEN Q3$(I)="00"
      :CONVERT Q3$(I)TOA9
      :STR(T2$(I),10,1)=BIN(A9)
```

Still no ECO' for the MB!

Regards  
John Baxi

2790 IF COS = "M" THEN V\$ = " : DATASAVE DC #0,  
" : ERROR 6050B 1120: GOTO 2760

## C INSTALL

4/9/93 ADDED TESTS FOR 3 BYTE ADDRESSES FOR INPUT & OUTPUT ADDRESSES.

```
175 DIM A$(1): A$(1) = STR$(SB(-), 1, 1): IF A$(1) <> HEX(02) THEN 177: PRINT AT(22, 0, 79);  
HEX(07 0E); " ADDRESS "; A$; " IS A 3 BYTE ADDRESS. THREE BYTE ADDRESSES NOT  
SUPPORTED. ": GOTO 140
```

```
177 PRINT AT(22, 0, 79)
```

```
215 A$(1) = STR$(SB(-), 1, 1): IF A$(1) <> HEX(02) THEN 217: PRINT AT(22, 0, 79);  
HEX(07 0E); " ADDRESS "; A$; " IS A 3 BYTE ADDRESS. THREE BYTE ADDRESSES  
NOT SUPPORTED. ": GOTO 180
```

```
217 PRINT AT(22, 0, 79)
```

3/9/94 UPDATED TO LOAD CS/386 O/S 1.30 FROM EITHER 1 1.2M OR 3 360K DISKETTES  
CHANGING NUMBER OF FILES INCLUDED W/ O/S FROM 36 TO 70.

```
640 REM % FF ENCOUNTERED: F = F + 1: IF C3 < 3 THEN 680: IF F = 3 THEN 680
```

```
642 C1$ = "CP": IF C3 > 3 AND F = 1 THEN C1$ = "CCLOC"
```

```
643 IF C3 > 3 AND F = 2 THEN C1$ = "CDSCFIG"
```

```
650 PRINT AT(17, 9, 79): PRINT AT(14, 9, 71); " MOUNT SYSTEM DISKETTE "; F + 1; " IN  
INPUT DRIVE, THEN HIT ANY KEY. ": KEYIN Z1$: PRINT AT(14, 9, 79): PRINT AT(13, 9, 79)
```

```
655 GOSUB 10(C1$, 1): IF T <> 0 THEN 660: IF F = 1 THEN PRINT AT(13, 9, 71); " SECOND  
DISK NOT MOUNTED. "
```

```
657 IF F = 2 THEN PRINT AT(13, 9, 71); " THIRD DISKETTE NOT MOUNTED. ": GOTO 650
```

```
660DEL RESTORE LINE 850: IF C3 > 3 AND F = 1 THEN RESTORE LINE 882882: IF C3 > 3  
AND F = 2 THEN RESTORE LINE 888: GOTO 410
```

```
670 REM % FF ENCOUNTERED: X = 0: RESTORE LINE 860:DEL GOTO 410
```

```
880 REM CS/386: DATA TO, FILE LIST (DELETED C$SYSMPB, STARTD, START, C$MENU,  
C$MOVE1, CPSTAT, & C$INSTALL) (ADDED C$ASRJV1 + C$HQ300V0)
```

LINE 890 CHANGES TO 882 + 884 (DELETIONS FROM 880 ADDED) (ADDITIONS TO 880 LOST)

LINE 888 + 890 ADDED FOR CS/386 Disk 3



CHANGE MESSAGE ON LINE 280 TO BE MORE DESCRIPTIVE.

```
280 PRINT HEX(07 0E 06); AT(22,0,79); R#; " CVP, CMVP NOT ON INPUT DISK.
```

```
INSERT CORRECT DISK OR CHANGE ADDRESS." : GOTO 140
```

3/10/94 UPDATED TO LOAD CS/TURBO O/S 1.30.01 FROM<sup>3</sup> 360Ks OR 1.2M DISKETTES  
CHANGING NUMBER OF FILES INCLUDED FROM 65 TO 75.

```
230 D1#, D4# = "CMVP": GOSUB '20(D1#): IF T=0 THEN 240: C3=3: E2=E1-B: IF E2 >  
450 THEN C3=4: IF E2 > 1000 THEN C3=5
```

```
235 IF C3 <> 4 THEN 237: PRINT AT(8,0,79); "INPUT DISK CONTAINS CS/386 O/S. ";  
LINPUT "INSTALL Y/N? " - B# : IF B# = "Y" OR B# = "y" THEN 295: GOTO 290
```

```
237 IF C3 <> 5 THEN 240: PRINT AT(8,0,79); "INPUT DISK CONTAINS CS/TURBO O/S. ";  
LINPUT "INSTALL Y/N? " - B# : IF B# = "Y" OR B# = "y" THEN 295: GOTO 290
```

```
290 B# = " " : PRINT AT(8,40); HEX(0E 06); " (CVP(Y), CMVP(M), CCS(386(W),  
CS/TURBO BASIC(T))": PRINT AT(8,0); HEX(0E 06); LINPUT "INSTALL WHICH SYSTEM "  
- B# : $TRAN(B#, "CvMwT") R: C3 = POS("CVMWT" = B#)
```

```
295 ON C3 GOTO 300, 310, 320, 325, 327: PRINT HEX(07): GOTO 290
```

```
325 D1# = "CGEN.386": GOTO 330
```

```
327 D1# = "C2236MXF"
```

```
640 REM %. END ENCOUNTERED: F = F + 1: IF C3 < 3 THEN 680: IF F = 3 THEN 680
```

```
660 RESTORE LINE 850: IF C3 = 4 AND F = 1 THEN RESTORE LINE 882: IF C3 = 5 AND  
F = 1 THEN RESTORE LINE 910: IF C3 = 4 AND F = 2 THEN RESTORE LINE 888: IF C3 = 5
```

```
AND F = 2 THEN RESTORE LINE 930: GOTO 410
```

```
765 ON C3 GOTO 770, 775, 780, 785, 787: STOP #
```

```
787 RESTORE LINE 900, X+1: H# = "CS/TURBO BASIC-2": C3 = 5: RETURN
```

```
900 REM .CS/TURBO: DATA DATA: 75 75, "C", "CBOOT", "CMVP", "C2236MXF",  
"C22CIHS", "C22CISS", "CMXEP", "CD6Z", END
```

```
910 REM .CS/TURBO DISKETTE 2: DATA " " 920 DATA " " ; END " 31 FILES
```

```
930 REM .CS/TURBO DISKETTE 3: DATA " " 940 DATA " " ; END " 36 FILES
```

CHANGED COPY DATE, ON MENU HEADING TO 1994, LINE 120.

ADDED UPDATE REMARK. LINE 12.

3/2/95 WHEN RUNNING @INSTALL ON A TURBO, ALWAYS COMES UP WITH @MVP NAME ON SOURCE = @386. ALSO HAD A USELESS PRINT COMMAND.

ON LINE 385 DELETED: D4@="@386":PRINT:



ISS.000M

CHANGES MADE FROM 5.5 TO 5.5+

DELETED CPU RESTRICTION (ORANGE - DELETED CODE)

9/29/89

```

420 A# = #PSTAT(#PART): IF STR(A#,9,1) = "M" THEN S3=4: ELSE S3=3: IF S3=4 AND
STR(A#,10,1) < HEX(17) THEN 1440: IF S3=3 AND STR(A#,10,1) < HEX(19) THEN 1460:
#G10/005(7601,A#): A# = A# AND HEX(10): IF STR(A#,1,1) = HEX(10) THEN S0=80: ELSE S0=
64: SELECT PRINT 005(S0): S = SPACEK

```

ISS.2055

ALLOWS ADDRESSES BEYOND Dx5 (CHANGES MADE FROM 5.5 TO 5.5+)

(ORANGE - DELETED CODE)

9/29/89

```

8875 DEFFN'Z05(R,W3#,R1): Q# = " ": MAT SEARCH" 310320330350360370B10B20B30B50
360B70D10D11D12D13D14D15D20D21D22D23D24D25D30D31D32D33D34D35P50D51D52D53D54
D55D60D61D62D63D64D65D70D71D72D73D74D75", = STR(W3#, , 3) TO WZ# STEP 3: IF R1 <> 0
AND R1 <> 1 OR R < 0 OR R > 15 THEN Q# = "X": IF W3# = "340" THEN 8895: $TRAN(W3#,
"ABBCcDdEeFf")R: IF POS("DB3" = W3#) > 0 AND POS("123567" = STR(W3#,2)) > 0
AND VER(STR(W3#,3), "H") > 0 THEN 8885: IF IF WZ# = HEX(00 00) THEN Q# = "I":
GOTO 8905

```

```

8885 IF POS("3B" = W3#) = 0 OR POS("123" = STR(W3#,2)) = 0 OR STR(W3#,3) <> "0" THEN

```

```

8895: IF STR(W3#, , 1) = "3" THEN STR(W3#,3) = "1": STR(W3#, , 1) = "D"

```

```

8895 Q# = " "

```

```

8905 IF R1 = 0 OR Q# <> " " THEN RETURN: SELECT #R < W3# >: RETURN

```

```

8915 IF R1 = 0 OR Q# <> " " THEN RETURN: SELECT #R < W3# >: RETURN

```

ISS 2295

ADDED SUPPORT FOR CS/386 (CHANGES MADE FROM 5.5 TO 5.5+)

8/29/89

7167 R9#(2) = #PSTAT(1): IF STR(R9#(2), 9, 1) = "V" THEN 7179: IF STR(R9#(2), 9, 1) = "W"  
THEN RETURN: IF STR(R9#(2), 10, 1) > HEX(24) THEN RETURN: PRINT .....

## CHANGES FROM 5.5+ TO BAXI'S ~~REL~~ REL (5.3?)

ISS.2295*	<sup>REL</sup> 25, 29	(24-33)
ISS.000A	132-134, 136, 137	(131-139)
ISS.050S	190-194, 197-200	(189-201)
ISS.001M	205-207, 210, 213-218	(204-219)
ISS.002M*	514-524	(513-525)
ISS.081U	540, 553	(539-571)
ISS.090U	575, 579	(574-582)
ISS.REF1	644-647	(644-649)
ISS.205S*	651-654	(650-655)
ISS.001U	672, 683-692	(671-692)
ISS.000M	826-830, 833 42-52, 55	(819-832)
ISS.000E	872 188	(870-872)
ISS.J21U	853, 867 259, 272	(853-867)

\* CHANGE IN REL 5.5+

2<sup>ND</sup> SECOND SECTOR ON ALL <sup>OTHER</sup> FILES WAS DIFFERENT. APPEARED TO BE DUE TO VERSION. ALL FILES ON BAXI'S VERSION APPEAR TO BE VERSION 5-3 OR 5-2, OTHERWISE SAME AS REL 5.5.

BAXI CHANGES: SHOWS & ALLOWS ALL DISK ADDRESSES. RECOGNIZES TURBO.



MOVEFIL

4/7/93 ADDED TEST FOR 3 BYTE INDEX FOR INPUT & OUTPUT ADDRESSES. MESSAGE COMES UP "3 BYTE ADDRESSING NOT SUPPORTED".

50 DIM A\$(1)

INPUT ADDR

150 PRINT: INPUT: IF: IF: C1=: IF: PRINT: C1: GOSUB '105: A\$(1)=STR(S\$( ), 1,1): IF A\$(1)=HEX(02) THEN 155

152 IF Q\$=" " THEN 160: A3\$=" ": GOTO 130 was on 150

155 IF A\$(1) <> HEX(02) THEN 160: PRINT AT (14,10,); HEX(0E); " ADDRESS "; A1\$; " IS A 3 BYTE ADDRESS. THREE BYTE ADDRESSES NOT SUPPORTED. ": GOTO 130

OUTPUT ADDR

440 GOSUB '105(2,A2\$,C2): A\$(1)=STR(S\$( ),1,1): IF A\$(1)=HEX(02) THEN 445: IF Q\$=" " THEN 450: A5\$=" ": GOTO 410

445 IF A\$(1) <> HEX(02) THEN 450: PRINT AT (14,10,); HEX(0E); " ADDRESS "; A2\$; " IS A 3 BYTE ADDRESS. THREE BYTE ADDRESSES NOT SUPPORTED. ": GOTO 410



VAL 200 320 460 520 5200 5205 5300 5320 5330 5350 5550 5700 5710

SODIM A#(i) BIN 5360 5470 5505 5508 5700 5710

MOVE FILE UTILITY FAILS w/ MESSAGE "OUTPUT PLATTER FULL" IF GO BEYOND 65535.

INPUT D10 OUTPUT D34 MOVE ALL Y OVERWRITE N HALT/STEP

162 H# IS OVERWRITE Y/N

200 A# = D10 C1 = 0 Q# = 20<sub>HEX</sub>

5140 DEFFN'115 (N, S, SA, T)

5142 RETURN

200 I1 = VAL(STR(SA( ), 2)) IF AB(1) = HEX(02) THEN CHECK INDEX TYPE I1 IS INDEX SIZE OF INPUT DISK \*

220 GOSUB'130 (0, 1, C3#, A#, I1-1, A##, C1):

5180 DEFFN'130 (0, N, N#, T8, T9, SA, T) = N3# = ALL(00) : Q# = N : R3 = 0 : I = T8 TO T9

GOSUB'115 (N, I, SA, T)

5140 DEFFN'115 (N, S, SA, T) : : DATALOADBAT#N, (S)SA( ):

5142

5180 FOR J=1 TO 16: IF POS(SA(J, 1) <> HEX(00)) = 0 THEN 5220 IF I = 0 & J = 1 OR STR(SA(J, 1), 1) = HEX(21) THEN 5230

5230 NEXT J, 1:

5180

5190 N3# = IF POS(N# <> HEX(00)) = 0 OR 0 <> 0 THEN 5200

5200 R3 = -SGN(VAL(STR(SA(J, 1), 2, 1)) / 128) + 2:

STR(SA(J, 1)) = 1000001800670000 FILE ENTRY 24 103

IF STR(SA(J, 1), 1) = HEX(11) THEN R3 = -R3 (IS FILE SCRATCHED)

R4 = VAL(STR(SA(J, 1), 3), 2) R4 = 24 START SECTOR \*

R5 = VAL(STR(SA(J, 1), 5), 2) R5 = 103 END SECTOR \*

R1 = 5 : R2 = J

5205 GOSUB'115 (N, R5, SA, T)

5140 DEFFN'115 (N, S, SA, T) DATALOADBAT#N, (S)SA( )

5142

READ LAST SECTOR IN FILE FOR USED FIELD

A0/00 4B/00 0000



5205 R6 = VAL(STR(S#( ), 2), 2) READ BYTES 2 & 3 OF ~~USED~~ LAST SECTOR FOR USBP FIELD

5210 Q# = " " : S8 = J S9 = I I = T9

5220 J = 16

5230 NEXT J, I

220 IF N3# = "ESPAND01" THEN C9# = "Y" : IF QA# = "N" & C9# = "Y" THEN 250

240 Z# = C1#, C2#, C3# = N3#

242 A9 = S9

270 P9# = C1# : IF Z = 1 THEN 280

280 C# = C1#

290 # OPEN #1, #2 : "GETTING FILE PARAMETERS" : IF C1# <> N3# OR QA# <> " " OR P9# =

C1# THEN GOSUB '130 (1, 1, C1#, #, I1-1, A1#, C1)

5180 DEFFN '130 (O, N, N#, T8, T9, S#, T)

N3# = "00" : QA# = "N" : R3 = # : I = T8 TO T9

GOSUB '115 (N, I, S#, T)

5140 DEFFN '115 (N, S, S#, T) DATA LOAD BAT#N, (S) S#( ) LOAD SECTOR #

5142 RET

5180 FOR J = 1 TO 16 IF I = # & J = 1 : 5230

5230 NEXT J, I

5180

5190 N3# = S#(J, 2)

5200 R3 = -SGN(VAL(STR(S#(J, 1), 2, 1)) / 128) + 2

IF STR(S#(J, 1), 1) = HEX(11) THEN R3 = -R3 IS FILE SCRATCHED

R4 = VAL(STR(S#(J, 1), 3), 2) R4 IS START SECTOR FOR C-MXE#

R5 = VAL(STR(S#(J, 1), 5), 2) R5 IS END SECTOR FOR C-MXE#

R1 = S, R2 = J



# MXE CHRONOLOGICAL CHANGES

MVP 2.7/MXE 2.66 JULY 1986

FIXED

1. \$BREAK! NOW WORKS PROPERLY.

ANOMALY: HALT/STOP OF A LINPUT COMMAND MAY NOT PROPERLY CLEAR DISPLAY AFTER RETURN.

MVP 3.0/MXE 3.0 JULY 1987

FIXED

1. DOWNLOAD OF MXE CODE DURING MASTER INIT COULD FAIL IF MXE FILE CONTAINED DATA BEYOND TRAILER RECORD.
2. \$BREAK! TO SUSPEND A PARTITION COULD INADVERTENTLY FAIL IF THE \$BREAK WAS EXECUTED SHORTLY AFTER A KEYIN STATEMENT WITH 2 LINE NUMBERS.
3. UNDERLINES IN A LINPUT FIELD WRAPPING FROM 1 LINE TO NEXT WOULD SOMETIMES REMAIN AFTER EXEC OF LINPUT.

MVP 3.1/MXE 3.0Z SEPT 1987

FIXED

1. IN TC MODE, IF SENDING 5 OR 6 BIT CHARACTERS, SOMETIMES A SHIFTED CHARACTER WOULD NOT BE SENT UNTIL ANOTHER CHARACTER WAS SENT.

2. IN SOME INSTANCES, PRINTING TO ADDR 204 OF A 2436WP WOULD STOP & WOULD NEED TO REBOOT TO CORRECT.

MVP 3.2.7/MXE 3.0A

SLOWDOWN ON LIST TO 204 WITH DRIVER ON

MVP 3.3/MXE 3.0XX FEB 1989

FIXED

1. SYSTEM HESITATED UP TO 1 SEC WHEN TYPING IN BEFORE DISPLAYING & THEN WOULD DISPLAY IN BURST.
2. AFTER INPUT SCREEN, SYSTEM COULD LOSE CHARACTERS IF RAPIDLY ENTERED ON A KEYIN.
3. SYSTEM LOCK WOULD LOSE TIME IF PRINTING TO PORT 4'S LOCAL PRINTER ON MXE 1.

ENHANCE: PERFORMANCE SIMILAR TO MXD. 19.2 BAUD ALLOWED FOR TC

MVP 3.4/MXE 3.10 MAY 1990

FIXED

1. SYSTEM HESITATES PRINTING TO ADDRESS 204.

ENHANCEMENT: PERFORMANCE SIMILAR TO MXD. ALLOWS 38.4 BAUD.

CPSTAT

11/19/92

ADDED LINE 47 TO DISPLAY ON SCREEN:

PRESS SF'12/NEXT FOR NEXT SCREEN, SF'13/PREV FOR PREVIOUS SCREEN.

3/21/95 FAILED ON NON-386 CPU: w/ ERROR S19. POINTING AT #CPU FOLLOWING CONVERT.

FIX: 28

:REM % CONVERT #CPU TO R#, (##):



Package Subject: Bugs with SCSI Config

Item Title: Bugs with SCSI Config

John,

Found a couple of minor bugs with the SCSI Config program. Sorry to bother you with this but if you still are working with it they are as follows:

1. Ran the SCSI Config program through twice while reviewing the Utility Mnl. On the 2nd pass after creating a new configuration, the system put the password entered in the 1st configuration on the dotted line where the number of sectors for surface 1 would be entered. The password was 'SYSTEM'. It looked something like this:

```
          SETUP SCSI CONFIGURATION
"
"
```

```
SECTORS REMAINING :    620178
AMOUNT FOR SURFACE 1 SYSTEM
```

2. As I mentioned on the phone the system will not accept odd numbers. It just beeps at you. 11/19/92 CHANGED LINE 1280 WHICH PRINTS OUT SECTOR PROMPT TO:

"SECTORS TO ASSIGN (EVEN #'S ONLY) TO ADDRESS"

3. If last entry when assigning sectors to surfaces exceeds available space the system jumps back to the screen where it asks if you want a Master or Slave. Should allow you to correct your entry. 11/19/92 ADDED TEST TO LINE 1300 TO DETERMINE IF # EXCEEDS 'REMAINING SECTORS. : IF B > TI THEN 1315 MOVE GOSUB #202 ON END OF LINE 1320 TO NEW LINE 1315. ADD PRINT AT (23,0,32) TO LINE 1280 TO CLEAR ERROR MESSAGE AFTER EACH ENTRY. CORRECTED ERROR MESSAGE TO "AMOUNT EXCEEDS AVAILABLE SECTORS."

4. If you enter 'NO' when asked if all entries made are acceptable you are sent back to where the system asks you the 'number of surfaces' again. That is fine but when you get to the point where you start assigning sectors to each address, available sectors equals 0 and any entry takes you back to problem 3. The utility needs to show the total disk space for that drive again. 11/19/92 RESET AVAILABLE SECTORS BACK TO FULL AVAILABILITY ON DRIVE BY ADDING TO LINE 1410; TI=T.

5. As I mentioned on the phone I think it is important to show the drive ID # on the screen where each address is displayed along with the catalog info. It also appears that any drive already configured may be damaged if a new configuration is created even if the addresses don't conflict. After I created the new configuration I had some strange information showing on the screen showing the catalog information. This could become a major problem when replacing 1 bad drive for a customer with more than 1 drive. Some of the catalogs look like they may be ok but it seems to be different from before. For example 2 of the surfaces had an index size followed by an &: 30& and 3& Is that legitimate? If I don't hear back from you I will give you a call later in the week.

Best regards,  
Mike

11/19/92 CHANGE LINE 1230 TO READ, "PLEASE ENTER NO. OF ADDRESSES TO ASSIGN TO DRIVE".

12/4/92 Added 'FN/TAB - EXIT' TO ALL SCREENS. SEE LINE 1020, 1165, 1225, 1440  
ADDED "WARNING" TO APPLY SCREEN. LINE 1440.

12/17/92 CHANGE GOTO ON END OF LINE 1410 FROM 1170 TO 1150 TO REMOVE ALL PREVIOUS SECTORS VALUES FROM ADDRESSES SHOULD YOU RESPOND NO TO 'CONFIGURATION ACCEPTABLE?'

MADE ENTRY < 100 UNACCEPTABLE FOR SECTOR SIZE. LINE 1290 ADDED IF B < 100 THEN 1280.  
~~CHANGED 1280 IS LENGTH OF B# MUST BE 3 CHARS. B# = SECTOR SIZE.~~

12/21/92 REMOVE Y DEFAULT ON ALL ENTRIES ACCEPTABLE. LINE 1390

12/22/92 IF 0 ENTERED AS SECTOR VALUE ASKS YOU IF ENTRIES ACCEPTABLE.

~~LINE 1290 ADDED IF B = 0 THEN 1390~~

CHANGED LENGTH OF B# TO 1 CHAR. LINE 1280.

EXTENDED MESSAGE WHEN ASKS FOR CONFIGURATION FILE NAME. LINE 1540.

2/16/93 CONFIGURATION SPELLED WRONG ON LINES 160 & 350, CORRECTED.

# PRELIMINARY CHANGE

"SCFG3/11" M4 TURBO-D11 ESCSICFG

3/1/93 IF YOU HAVE ASSIGNED ALL AVAILABLE SECTORS, BUT STILL HAVE ADDRESSES REMAINING THAT WERE INDICATED TO BE USED,  $\emptyset$  WOULD NOT BE ACCEPTED. HAD TO KEY FN/TAB TO EXIT & RESTART.

Fix - IF  $\emptyset$  SECTORS IS ENTERED, WILL END SECTOR ASSIGNMENT PROCEDURE & ASK IF "ALL ENTRIES MADE AND ACCEPTABLE?"

REMOVE IF LEN(B#) < 1 THEN 1280 FROM END OF LINE 1280  
ADD 1280 : IF B =  $\emptyset$  THEN 1390

- ADDRESS SPELT W/ TWO AA'S (ADDRESS) ON LINE 670. CORRECTED.

3/10/93 IF SAVE CONFIG & RERUN SETUP, PASSWORD SHOWS UP WHEN ON SECTOR ASSIGNMENT FIELD.

1520 GOSUB 602(" ", " "): B# = ALL(20): PRINT \_\_\_\_\_

- SYSTEM IS PASSWORD. CHANGED TO MAKE SYSTEM PASSWORD THE PASSWORD.

ADD 100 DIM V3#  
CHANGE FROM 1510 IF B# = "SYSTEM" THEN 1520: PRINT HEX(07);: GOTO 1500  
1510 V3# = DATE: DATE = V3# PASSWORD B#: ERROR PRINT HEX(07);: GOTO 1500

3/11/93 IF MORE THAN 15 ADDRESSES ARE ASSIGNED, THE FIRST SLAVE ADDRESS IS LOST AND ALL ADDRESS AFTER IT MOVE UP 1.

CHANGE LINE 1340 IF X=15 THEN DO: P=P+4: STR \_\_\_\_\_ : P=P+4: STR \_\_\_\_\_ = "END": P=149: ENDDO  
TO 1340 : : : : : P=153:

CHANGE LINE 1350 IF X=16 THEN DO: P=149: STR(A#( ), P, 4) = "scsi": P=P+4: R2=48: ENDDO  
TO 1350 IF X=16 THEN DO: V=149: STR(A#( ), V, 4) = "scsi": P=P+4: R2=48: ENDDO

- INT X71 ON LINE 660 OR 760. PRINT IMAGE NOT BIG ENOUGH FOR S. ADDED 1 #  
TO CONVERTS TO L# ON LINES 660, 670, 720, 760 OVER

3/11/93 WHEN EXPAND DISPLAY CATALOG MAXIMUM FIELD DROPS LEAST SIGNIFICANT  
FIELD. 65024 SHOWS AS 6502. ADD ~~LEFT~~ # TO LEFT OF CATALOG MAX  
FIELD ON LINE 7920.



SCSICFG 16TH ADDRESS GETS WRITTEN OVER

LOAD CONFIG N  
M OR S M  
ADDR'S 29

SECTORS 1-F 60000  
71 50000  
72 40000

ADDR 15 60000 HALT/RET

1280 PRINT  
1290 CONVERT <sup>SECTORS</sup> B\$ TO B IF = 0 END, IF < 100 REPEAT

1300 C=B MAKE SURE SECTOR # DOES NOT EXCEED REMAINING + IS ODD

1320 SZ = SECTORS USED + 2 (GOSUB) 840002 14 x 60000 = 840000

8070 DEFFN '600(D) D = SZ 840002

8080 G\$ = BIN(D, 4): F\$ = ALL(20) G\$ = 000CD142 F\$ = 60EA0000

8090 STR F\$ = G\$

1320 STR(A\$( ), P, 4) = F\$ AS A RESULT G\$ = 000CD142 F\$ = 000CD142

P = 117

P = P + 4 P = 121

GOSUB '600(C) C = 60000

8070 DEFFN '600(D) D = 60000

8080 G\$ = BIN(D, 4): F\$ = ALL(20) G\$ = 0000EA60 F\$ = 42D10C00

8090 STR F\$ = G\$ REVERSE G\$ = 0000EA60 F\$ = 60EA0000

1320 STR(A\$( ), P, 4) = F\$ P = 121

SZ = SZ + C UPDATE SECTORS USED SZ = 900002 C = 60000

B(X) = SZ B = 60000 X = 15 ADDR

T1 = T1 - B UPDATE REMAINING SECTORS

NO 1330 IF X < 15 THEN DO: P = P + 4: ENDO X = 15 P = 121

YES 1340 IF X = 15 THEN DO: P = P + 4 P = 125

STR(A\$( ), P, 4) = HEX(FF FF FF FF)

P = P + 4 P = 129

STR(A\$( ), P, 3) = "END": P = 149: ENDO P = 149

\*\*\* P = 153

NO 1350 IF X = 16 THEN DO

NO 1360 IF X > 16

1370 IF X > 0 THEN DO

PRINT FIELDS RZ = 8 A = 7 X = 15

1380 NEXT X

1270 PRINT "SECTORS REMAINING" TI 426202

1280 INPUT SECTORS FOR ADDR 16 50000

1290 CONVERT B# TO B, IF  $\emptyset$  END, <100 REPEAT

1300 C=B MAKE SURE DOES NOT EXCEED REMAINING SECTOR & NOT ODD

1320 (C=50000) (T=1326202) (A(X)=50000) (SZ=90002) GOSUB

8070 DEFFN'600(D) (D=900002)

8080 G# = 000DBBAZ F# = 60EA0000 CLR F#

8090 STR F# = G# REVERSE G# = 000DBBAZ F# = AZBBDDDD

1320 STR A#( , P, 4) = F# (P=149)

P = P + 4 (P=153)

GOSUB'600(C) C=50000

8070 DEFFN'600(D) D=50000

8080 G# = 0000C350 F# = AZBBDDDD

8090 G# = 0000C350 F# = 50C3DDDD

1320 STR (A#( , P, 4) = F# (P=153)

SZ = SZ + C UPDATE SECTORS USED

SZ = 950002 C = 50000

B(X) = SZ B = 50000 X = 16

TI = TI - B UPDATE REMAINING SECTORS

NO 1330 IF X < 15

NO 1340 IF X = 15

YES 1350 IF X = 16 (P=149) ★ V = 149

STR (A#( , P, 4) = "secs." V

P = P + 4 (P=153) ★★ ★

RZ = 48

NO 1360 IF X > 16

1370 IF X >  $\emptyset$  THEN DO PRINT FIELDS RZ=48 A=7 X=16

1380 NEXT X

1270 PRINT SECTORS REMAINING 376202

1280 INPUT SECTORS FOR ADDR 17 40000

1290 CONVERT B# TO B IF  $\emptyset$  END, IF < 100 REPEAT

1300 C=B C=40000

1320 C=40000 T=1326202 A(X)=40000 A=7 X=17 GOSUB(SZ) SZ=95002

8070 DEFFN'600(D) D=95002

8080 G# = 000E7BFZ F# = 50C30000

8090 F# - G# Rev G# = 000E7EF2 F# = F27E0E00

1320 STRA\$( ), P, 4) = F# P = 153

P = P + 4 P = 157

GOSUB '600(C) C = 40000

8020 DEFFN '600(D) D = 40000

8080 GA = 00009C40 F# = F27E0E00

8090 GB = 00009C40 F# = 409C0000

1320 STR(A\$( ), P, 4) = F# P = 157

S2 = S2 + C UPDATE SECTORS USED S2 = 990002 C = 40000

B(X) = S2 B = 40000 X = 17

T1 = T1 - B UPDATE REMAINING SECTORS

NO 1330 IF X < 15

NO 1340 IF X = 15

NO 1350 IF X = 16

YES 1360 IF X > 16

P = P + 4 P = 161

1370 PRINT

1380 NEXT X

1270 PRINT SECTORS REMAINING

1280 INPUT SECTORS FOR ADDR 18 30000

```

8000 REM %t.Apply Changes to DS cabinet: REM %Code @ 8000-8990 will be a separate
module saved scrambled as @DSCFIG8
8010 DEFFN'10: IF #TERM(>)THEN 10: D#=SELECT #2: STR(D#,2,1)=STR(D#,2,1)AND HEX(F#):
STR(D#,3,1)="0": D1#=SELECT #1: PRINT HEX(03): PRINT AT(0,0);"ALL DATA ON W
INCHESTER DRIVES IN THE UNIT BEING CONFIGURED SHOULD BE BACKED UP!!"
8012 PRINT AT(23,55);"FN/TAB - Exit";AT(1,0);"Apply changes to cabinet ";D#;" fr
om file on ";D1#
8015 X=4: IF STR(E#(1,4,2))>"3D"THEN 8020: GOSUB '50(HEX(0E),"You need DS proo
vel 4 to apply changes"): KEYIN K#: GOTO 10
8020 PRINT AT(2,0);HEX(0E);"Enter PASSWORD to apply changes": Z#=" ": PRINT AT(2
,40);"LINPUT -Z#: V3#DATE: DATE=V3#PASSWORDZ#: ERROR PRINT HEX(07): X=X-1: IF
X>0THEN 8020: GOTO 10
8025 F#="8DEFAULT"
8030 PRINT HEX(07): PRINT AT(2,0,80); LINPUT "Configuration file name "-F#
8040 LIMITS T#1,F#,W1,B,C,E: IF E<>2THEN 8030: IF C<>8THEN 8030
8045 PRINT AT(1,0,80);HEX(0E);"A P P L Y I N G   U P D A T E   T O   D S   P R
O G R A M": D1=1
8050 REM %Drive configuration sector: REM .Bytes.001-032 Drive constants: REM .
001-007 Reserved for alternate sectoring: REM .008-010 reserved for future use
REM .011-013 "TBO": REM .014 binary count of platters in drive: REM .015 st
platter eg 01, 05, 41
8060 REM .016.# heads/drive: REM .017 cylinder to start RWC: REM .018 # cylinder
s for alternate sectors: REM .019 drive switch setting: REM .020-032 reserved fo
;

```



CSCAPEB

3/15/93 IF TYPE IN WRONG ADDR, A NON-SCSI ADDR, OR NO TAPE DRIVE IS FOUND,  
FN/TAB MESSAGE TO EXIT DISAPPEARS.

FIX: CHANGE GOTO 150 ON LINES 150, 180, & 185 TO GOTO 130.

# CSCTAPER

3/15/93 IF TYPE IN WRONG ADDRESS FOR TAPE, A NON-SCSI CONTROLLER ADDR, ON TAPE IS NOT RECOGNIZED, GET ERROR BUT FN/TAB MESSAGE TO EXIT IS GONE.  
FIX: CHANGE GOTO 90 ON LINES 90, 120, + 125 TO GOTO 70.

3/16/93 FN/TAB DISAPPEARS WHEN TAPE DIRECTORY LOADED.  
FIX: CHANGE PRINTAT STATEMENTS ON LINES 7040-7060 FROM (23,79)  
TO (23,0,60).

C245MVPB

2/20/98 ADDED NEW LINES  
9115 DATA "ELOCK", "DISPLAY CLOCK", "P"

© TO.CRE#

12/23/92 CHANGE SCREEN HEADING: LINE 35

FROM:

TO: CREATING/MOVING A SELECTED LIST OF FILES - (C) COPR. WANG LABS.,.....

REMOVED X FROM 4 SAVE OPTIONS. LINES 521, 522, 523.

4/7/93 ADDED TEST FOR 3 BYTE INDEX FOR INPUT ADDRESS. NOT SUPPORTED.

55 VERIFY T#2(0,0)A: ERROR A=1: IF A=1 THEN 60

57 DIM B\$(16), A\$(1): DATALOAD BA T#2, (0) B\$( ): A\$(1) = STR(B\$( ), 1, 1): IF  
A\$(1) <> HEX(02) THEN 60: PRINT AT(2, 25); HEX(0E); " ADDRESS "; D\$; " IS A 3  
BYTE ADDRESS. NOT SUPPORTED. ": GOTO 50

6/10/93 CHANGED LINE 55 TO CIRCUMVENT TURBO BUG<sup>1.18</sup> WHERE VERIFY T(0,0) TO 2275  
VERIFY WHOLE DISK & RETURNS I98 ERROR.

55 VERIFY T#2, (0,1)A: ERROR A=1: IF A=1 THEN 60



# 3

# Tuesday April 1990

3 7:00 1<sup>57</sup> MISSING 1155 1178 59  
 7:30 \* TBO.HASH 8 3/14/90  
 2/26/90 8:00 TBO.FNAM 9  
 8:30 TBO.SDC1 39 42  
 9:00 - TBO.ZAP 50 53  
 9:30

4 10:00 2<sup>57</sup> MISSING 1154 1178  
 10:30 TBO.FNAM 9  
 11:00 TBO.HASH 8  
 11:30 - TBO.LDT 10 11  
 12:00 TBO.SDC1 39 42  
 1:00 TBO.ZAP 50 53  
 1:30

7 2:00 3 58 FILES 1149 SECTORS  
 2:30 .....E. 65  
 3:00 C CLOCK 68 74  
 3:30 CDS.TEST 21  
 4:00 E 10  
 30 PAT2.7 13  
 5:00 PATCH5 13  
 STARTTBO 15 16

7:00 TBO.CKDS 19  
 7:30 → TBO.COPS 33 34 4/10/88  
 8:00 ~ TBO.CRF3 52 54  
 8:30 ~~TBO.DATA 4~~  
 9:00 \* TBO.E 10 8/12/82  
 9:30 \* TBO.E1 65 8/10/82  
 10:00 TBO.FINF 5  
 10:30 TBO.FNAM 9  
 11:00 TBO.HASH 8  
 11:30 \* TBO.LDT 11 4/29/88  
 12:00 - TBO.SDC1 37 42 4/13/88  
 1:00 TBO.SECT 11  
 1:30 TBO.ZAP 48 53  
 0 TBO.PSTAT 12  
 2:30 TEST9000 5  
 3:00

5 3:30 6 57 FILES 1152 SECTORS  
 4:00 TBO.HASH 8  
 4:30 TBO.LDT 10 11  
 5:00 TBO.SDC1 39 42  
 - TBO.ZAP 48 53  
 TBO.FNAM 9

2 7 58 FILES 1165 SECTORS  
 3/14/90 TBO.SDC1 39 42  
 TBO.FNAM 9  
 ✓ TBO.ZAP 52 53

February 1990							April 1990						
S	M	T	W	T	F	S	S	M	T	W	T	F	S
				1	2	3	1	2	3	4	5	6	7
4	5	6	7	8	9	10	8	9	10	11	12	13	14
11	12	13	14	15	16	17	15	16	17	18	19	20	21
18	19	20	21	22	23	24	22	23	24	25	26	27	28
25	26	27	28				29	30					

April 1990  
 S M T W T F S  
 1 2 3 4 5 6 7  
 8 9 10 11 12 13 14  
 15 16 17 18 19 20 21  
 22 23 24 25 26 27 28  
 29 30

2

EverReady® E717 by Keith Clark

6 4 55 FILES 1118 SECT  
 - ECLOCK 68 74 1/30/89  
 - STARTTBO 15 16 10/6/89  
 \* TBO.CKDS 19 10/9/86  
 - TBO.CRF3 52 54 3/14/88  
 \* TBO.FINF 5 8/14/89  
 TBO.FNAM 9  
 TBO.HASH 8  
 - TBO.LDT 8 11 10/24/89  
 TBO.SDCI 39 42  
 \* TBO.SECT 11 10/8/89  
 TBO.ZAP 48 53  
 X TBO.PSTAT 12

8 5 52 FILES 1005 SECTIONS

March 1990 May 1990  
 S M T W T F S S M T W T F S  
 .....E.  
 1 2 3 4 5  
 9 10 11 12  
 17 18 19  
 25

Sunday  
 April  
 1990

ECLOCK 68  
 E 10  
 - STARTTBO 15 16  
 - SYS TBO 10 12 11/20/87  
 9:00 \* TBO.BGIO 31 12/30/87  
 9:30 \* TBO.ADI 13 11/19/87  
 10:00 TBO.CKDS 19  
 10:30 TBO.CMPS 33 34  
 11:00 TBO.CRF3 52 54  
 11:30 TBO.E 10  
 12:00 TBO.EI 65  
 1:00 \* TBO.FIND 43 11/27/87  
 1:30 TBO.FINF 5  
 2:00 TBO.FNAM 9  
 2:30 TBO.HASH 8  
 3:00 TBO.LDT 11  
 3:30 TBO.RENM 4  
 4:00 - TBO.SDCI 36 42  
 4:30 TBO.SECT 11  
 8:00 - TBO.XDAD 15 21  
 8:30 TBO.ZAP 48 53  
 9:00 - TBO.MDATA 9 19 11/18/86  
 9:30 TBO.PSTAT 12  
 10:00  
 10:30 6 57 FILES 1152  
 11:00  
 11:30  
 12:00  
 1:00  
 1:30  
 2:00  
 2:30  
 3:00  
 3:30  
 4:00  
 4:30  
 5:00

# CS/2200 Tool Box Option Updated

by Tyler B. Olsen

*[Editor's note: Tyler Olsen is the primary author of the CS/2200 Tool Box Option.]*

An updated version of the CS/2200 Tool Box Option, offered through the ISWU Software Library, is now available. The Tool Box Option is a subset of a collection of BASIC-2 programs that have been developed over the lifespan of the CS/2200 line to facilitate the development and maintenance of system software.

These programs are collected and maintained in a loose-leaf notebook. The programs are created by a variety of people but are collected and sometimes modified by one.

This update includes several changes from earlier versions. The cross-reference lister has been modified to include most BASIC-2 release 3 language enhancements. Disk prompting has been modified to include addresses of the Data Storage Cabinet. New features have been added to the disk catalog sort. Other program functions have been added, some enhanced.

## TOOL BOX GROUND RULES

My ground rules for developing and maintaining this collection are:

- The programs must reside on single diskette surface. The maximum number of sectors is 1232 on a 2270A-type drive. These criteria keep the number of programs to a manageable number.
- The basic operating instructions are to mount the diskette, then key RESET LOAD RUN RETURN and follow the prompts. There is documentation with the package which provides program abstracts in addition to operating instructions and descriptions of some functions.

- The functions must be useful to the CS/2200 system user. System diagnostics enable you to look at the hardware to see what is connected. Interrogation tools enable you to look at programs and data files. Documentation aids provide helpful tools for development or system maintenance. And some of the programs are for the creation, listing, and maintenance of telecommunications-formatted data files.

Tool Box Option diagnostic programs include "Monitor Partition Status," "What Disks Are On the System?" "Show Status Of Printers, TC Boards, and Terminal," "Analyze Disk Index," and "Show CRT Character Set and Keyboard Values." Interrogation tools include "Sort the Disk Catalog Area," "Disk Catalog Examination," and "File ZAP."

Documentation and development aids include "Sort the Disk Catalog Area," "Program or System Comparison," and "Cross-Reference Listing."

Other programs represent experiments that might be useful or of interest; e.g., "Analyze Disk Index," "Flow Chart Maker," "Data File Comparison," "Number Conversions," "Analyze \$GIO Statements," "@CLOCK" a clock and calendar, and many others.

Some of the programs contain one or more subroutines to perform a particular function. You may want to study these routines for code that could easily be included in your own system software. TBODISKS, for example, has code to look for all possible disk addresses; TBO.XDAD is used to determine the disk type; and @CLOCK creates large block letters and digits on the CRT.

If I find myself on an unknown system, I might run "Monitor Partition Status," "Show Status of Printers," and "What Disks Are On the System?"

If I'm looking at software packages, I am liable to use "Sort the Disk Catalog Area," "Cross-Reference Listing," or "Program or System Comparison." If used with all their bells and whistles, these three functions are adequate to document and maintain a software system.

#### A FEW PROGRAM DETAILS

The "Cross-Reference Listing" creates a cross-referenced listing of one or more programs from one or more disk surfaces. This program has been upgraded to process most of the language enhancements of release 3 of the Wang Multi-User BASIC-2 operating system.

The cross-reference program creates a listing of each program module that is page-numbered and titled. The first section is a decompressed listing of each program line; the summary section describes file references and variables used, prime subroutine locations, then statement and prime subroutine references.

The program uses tables that enable the user to describe BASIC-2 variables, prime functions, and file references with mnemonic descriptors of up to 16 characters. These mnemonic descriptors may be output in the margin of the listing as well as in the summary description. Use of the mnemonic descriptors is optional, but they can be extremely useful in documenting programs and systems.

"@CLOCK" uses the MXE board to provide a data and time clock. The clock has a large digital face and monthly calendar or message display area. A small file of operator-entered reminder messages is maintained for each terminal.

"Sort the Disk Catalog" sorts the items in the catalog index area for viewing. Data or program file names may be ordered by sector number, reverse sector number, or name. If ordered by name, either all or a common one- to eight-character ID may be requested. The display will show name and sector information only as with LIST DCT, but sorted. Alternatively, it will show the sector information with the descriptive remark or the image statement normally found as the first

line of any module. Provision is also made for output of descriptions of data files.

The sort has an option of looking for programs modified after a certain date. (I put a MM/DD/YY stamp on the front of each program module.)

Another option lets you look only for the presence or absence of an item from a list of names; this feature lets you see if all modules of a particular system reside on a particular surface. The list of names found is saved in a common array, which may be used as input by other programs. An exit is provided for this feature.

"Program or System Compare" compares two programs line by line and displays the differences. Alternatively, it will compare the modules on two different disk surfaces from a list of program names.

"What Disks Are On the System?" shows the cabinet type for each of the three disk controllers. It also shows the catalog index summary for each disk drive.

#### ORDERING INFORMATION

The CS/2200 Tool Box Option is available to all ISWU members. The package costs \$45 and is available on three types of diskette: 8-inch SSSD, 8-inch DSSD, and 5 1/4-inch.

To order the CS/2200 Tool Box Option (or if you have any questions), call the ISWU Software Library at (617) 967-1058. Have your ISWU membership number as well as a purchase order number ready. To order by mail, send a completed order form (following this article) along with a check or purchase order. Indicate the type of media you prefer on your order.

Allow three to four weeks delivery time for any software order. Express mail service is available for an additional \$25. Specify express mail service on your order if you wish to take advantage of it.



**Tyler Olsen**  
is a principal software engineer for the Wang Laboratories, Inc. (Lowell, MA) CS/2200 Product Group.





## Wang Micro-VP Tool Box Option Software

TYLER OLSEN

The "Wang Tool Box Option Software" is a loose-leaf collection of Wang 2200 Basic-2 programs that were developed to aid and assist in the development of Wang 2200 systems. The programs may be accessed from a simple command menu.

The 2200 Technical Support group has used these programs and thought they might be useful to the field as well.

### Programs are Unsupported

Programs are a Loose-leaf collection subject to change

Documentation is Reset, LOAD, RUN, RETURN  
and read the prompts on the CRT

The "Wang Tool Box Option Software" diskette includes:

Utilities and Program Development Aids

eg.

Cross-reference and Sort catalog area

Clock and calendar

Diagnostic Aids

Monitor Partition status

Prepare or list TC format data

Menu entry to system @MENU

An installation program to move files to a system platter

# Wang micro-VP Tool Box Option Software

## Index:

### Section I: Overview:

Disclaimer

Initial menus

Menu and system files of DATA statements.

### Section II: Abstracts for Tool Box Option Software:

### Section III. Operating instructions for selected utilities.





The menus activated by the utility "STARTTBO" operate off a list of DATA statements which normally overlay the code from 9000 - end. These DATA statements are of the form:

Bytes 1-8 Program module name

Byte 9 value = space if loading a program

= ] if loading a menu overlay (new code for statements 9000-end).

= c if saving COMMON variables, eg. disk addresses.

= @ if loading the standard @MENU program.

Bytes 10-70 are descriptive text to appear on menu line.

Line 15 of module "TBO UTIL" allows you to specify the most common disk surfaces on your system. When asked to specify a specific surface the operator may then key a single digit or a full disk address and EXEC.

```
0015 COM D1$(12)4: D1$()="310 B10 320 B20":REM/.up to 9 disks
```

```
9000 REM      ....!...V1....!....2....!....3....!....4....!....5....!....6....
9... DATA "Tool Box Option"
9... DATA "TBO UTIL]TBO Utilities
9... DATA "@CLOCK  Clock and calendar
9... DATA "TBO DIAG]Diagnostic Aids
9... DATA "TBO TCPL]Prepare or list TC format data
9... DATA "@SYS MVPB@MENU
9... DATA " "
```

Within the menu "TBO UTIL"

```
9000 REM TBO UTIL
9... DATA "Tool Box Option Utilities"
9... DATA "TBO.SDC0cSort the disk catalog area
9... DATA "TBO.CRF0cCross-reference listing
9... DATA "TBO.CMPScProgram or system comparison
9... DATA "@MOVEFIL @MOVEFIL
9... DATA "TBO.ZAP  File Zap
9... DATA "TBO.ANDFcAnalyze data file structure
9... DATA "TBO.XDC  cDisk catalog examination
9... DATA "TBO.SPV  cSearch programs for verbs
9... DATA "TBO.FLOWcFlow chart maker
9... DATA "TBO.RENMcRename data file
9... DATA "TBO.DMAPcMap disk for program call integrity
9... DATA "TBO.CMD  cData file comparison
9... DATA "TBO NOTE Notes on TBO utilities
9... DATA "TBO NUMC]Number conversions
9... DATA " "
9... DATA "TBO.CRF1 Crossref COMMON
9... DATA "TBO.CRF2 Crossref set-up
9... DATA "TBO.CRF3 Crossref mainline
9... DATA "TBO.SDC1 SORTCAT  mainline
9... DATA " "
```

Within the menu "TBO DIAG"

```
9000 REM TBO DIAG
9... DATA "Tool Box Diagnostics"
9... DATA "TBO.XDAD What disks are on the system?
9... DATA "TBO.CRT  CRT character set and keyboard values
9... DATA "TBO.STAT Show status of Printers, TC boards, & terminal
9... DATA "TBO.XASK Examine ASKACALL file
9... DATA " "
```

## II. Abstracts for Tool Box Option Utilities

**Function:** Cross-reference listing  
**Modules:** TBO.CRF0 - TBO.CRF1 - TBO.CRF2 - TBO.CRF3  
**Abstract:** One or more program files on any system disk may be listed on numbered and titled pages. For each program requested the listing consists of decompressed program statements followed by a cross-referenced listing of variables used, special function, and numbered statement references.

BASIC-2 commands operate on a program in the user partition:

**LIST** lists all lines.  
**LISTSD** lists a Section Decompressed.  
**LIST V** lists Variables used.  
**LIST '**  lists Special Function references.  
**LIST #** lists numbered statement references.

**Function:** Sort the disk catalog area  
**Modules:** TBO.SDC0 - TBO.SDC1

**Abstract:** Sorts the entries of the disk catalog into an ordered sequence. The listing may be in name sequence or in sector number, or reverse sector number order. Program or data file names are listed item by item in separate groups. Listings requested by name may include all or a subset of the files stored.

BASIC-2 command **LIST S DCT** lists all file names in hash sequence order.

**Function:** Program or system comparison  
**Modules:** TBO.CMPS  
**Abstract:** Program files on any system disks are compared line by line. Differences between the two files are displayed line by line. Two system disks may be compared by activating a list of DATA statement names.

BASIC-2 commands -- None.

**Function:** @MOVEFIL disk to disk file copy  
**Modules:** @MOVEFIL  
**Abstract:** This program from the standard BASIC-2 release copies program and data files from one disk surface to another.

**Function:** File Z A P  
**Modules:** TBO.ZAP  
**Abstract:** A single disk surface may be examined. The display shows the contents of a single sector in hexadecimal notation and in ASCII.  
1. Items in the catalog index area are flagged as AP (Active Programs), AData (Active Data).  
2. Data files are displayed with highlighted attribute bytes.  
3. Program files are displayed with highlighted RETURN codes.  
Caution, you have the ability to change any all data in a selected sector.  
BASIC-2 commands -- None.

**Function:** Analyze data file structure  
**Modules:** TBO.ANDF  
**Abstract:** This program analyzes a catalogued data file for structure. The output is a summary of the SAVE statements used to create the file.  
**Example** File name=@SYSFILE  
 File size = 32 Sectors.  
 File=@SYSFILE BASIC-2 data structure -- SAVE number 1  
 \$8 Scalar \$32 \$( 2)16 \$(16)8 \$(33)3 \$(15)13

**Example** File name=BSC\*010A  
 File size = 40 Sectors.  
 File=BSC\*010A BASIC-2 data structure -- SAVE number 1  
 \$( 256)16  
 File=BSC\*010A BASIC-2 data structure -- SAVE number 2  
 \$( 256)16  
 END OF FILE

**Function:** Disk catalog examination  
**Modules:** TBO.XDC  
**Abstract:** This program lets the operator 1) examine the disk catalog, 2), Examine a disk, or 3). Search disk for programs. I use 3) to locate programs by sector number when a disk index has been clobbered.

**Function:** Search programs for verbs  
**Modules:** TBO.ANDF  
**Abstract:** This program lets you search a list of programs for specific verbs. It was originally written to determine which kind of system was required to run a certain program. I have used it in two different environments:  
 1) to search systems for COM statements when trying to reduce memory requirements,  
 2) for places where string variables are set to quotable values, (useful when translating English menus to Spanish).

**Function:** Flow chart maker  
**Modules:** TBO.FLOW  
**Abstract:** This program was an early attempt to make flow charts from BASIC-2 code. It draws boxes around BASIC-2 code. Who cares? Someone who is required to submit flow diagrams of their program. The cross-reference program when used with REM% comments and variable annotation is far superior.

**Function:** Rename data file  
**Modules:** TBO.RENM  
**Abstract:** This program allows the operator to change a data file name.

Function: Map disk for program call integrity

Modules: TBO.DMAP

Abstract: This program traces the program map of a disk by keying off the program names and descriptions in load modules. In a TBO menu structured disk it will show which menu picks are further menus, which are programs, and which dead-ended", ie. programs that are non-existent on the disk.

Function: Data File comparison

Modules: TBO.CMD

Abstract: Data files on any system disks are compared byte by byte. Differences between the two files are highlighted and displayed sector by sector.

BASIC-2 commands -- None.

Function: Partition status

Modules: TBOPSTAT

Abstract: This program built from the SYSTEM UTILITY "@PSTAT" also shows the Device Table on one line toward the bottom of the screen. This addition is useful to determine if some device might be used or hogged by another partition.

Function: Disk status

Modules: TBO.XDAD

Abstract: This program displays the status of all disks on the system. It is a static display which goes through possible disk addresses on 10, 20, and 30

It shows Cur.end -Max., Access errors, disk not configured, disk is unavailable.

THIS IS THE STATUS OF POSSIBLE DISKS ON THE SYSTEM

1 310 Yes Cur.END=17414 Max.=38911 ! 2 B10.. etc.

3 350 Access error = 98

5

Function: CRT character set and keyboard values

Modules: TBO.CRT

Abstract: This program displays the full CRT character set. After the initial display, various keys can be depressed to determine the hex and ASCII values of keystrokes. (S.F.=hh) shows special function keys, ie those with an ENDI level.



Function: T.C. Board status

Modules: TBO.STAT

Abstract: This program displays the status of peripherals on selected device addresses. Printers are tested on 215, 216, 217, and 218. T.C. boards are tested on 6 addresses from 1A to 1F. For T.C. boards the program shows Ready/Not, B,C, or D, and memory available. The status of the operating system and user terminal configuration are also displayed.

BASIC-2 command LIST DT shows addresses configured, selected or hogged.

Function: Examine ASKACALL file

Modules: TBO.XASK

Abstract: This program displays the contents of the configuration file "ASKACALL" used in the ASC, BSC, and 2200/3270 emulations.  
N\$ is the name given the emulation by the operator,  
W\$ in non-3270 displays the modules loaded.  
Z\$ bytes 01-20 in hex -- are parameters for the microcode.  
bytes 21-64 in ASCII are responses to prompts.

## Tool Box Option Utility

### III. Operating Instructions for selected utilities

Sort the disk catalog area	2 pages
Clock and calendar	2 pages
Disk status	1 page.
T.C board, Printer and Terminal status	1 page.
LOAD RUN xxx	1 page.
Cross-reference listing	3 pages
Program or system comparison	2 pages
Analyze data file structure	1 pages
File Z A P	2 pages

**Abstract:** Sorts the entries of the disk catalog into an ordered sequence. The listing may be in name sequence or in sector number, or reverse sector number order. Program or data file names are listed item by item in separate groups. Listings requested by name may include all or a subset of the files stored.

**Modules:** TBO.SDC0 - TBO.SDC1  
**Equipment used:** CRT / keyboard and optional printer.

Operating instructions:

Display REQUEST NUMBER= 1 SORT DISK FILE CATALOG NAMES  
1 =310 2 =B10 3 =320 4 =B20

Prompt 1 Disk surface Key single digit or any valid disk address 1

Respond EXEC  
or digit and EXEC  
or disk address and EXEC

Display 1 =310

Prompt 2 Sort by 0=Name 1=Sector 2=-Sector (DEFAULT)=0 ?

Respond EXEC for sequence by Name  
or 0 and EXEC for sequence by Name  
or 1 and EXEC for sequence in Sector number order  
or 2 and EXEC for sequence in -Sector order, ie. last names entered

Prompt 3 Common Root ID ? Asked if Prompt 2 response was 0.

Respond EXEC to collect names of all files  
1-8 characters and EXEC to collect by common ID. A virgule ("/") in any position may be used for masked searching.

Display TYPE OF FILE NAMES TO SORT  
PROGS+DATA A S ALL  
PROGRAMS AP SP P=AP+SP  
DATA AD SD D=AD+SD

Prompt 4 Category Active Scratched (DEFAULT)=AP?

Respond EXEC for sequenced list of Active Program file names.  
or AP and EXEC for Active Programs.  
or SP and EXEC for Scratched Programs.  
or AD and EXEC for Active Data file names.  
or SD and EXEC for Scratched Data file names.  
or A and EXEC for Active Program and Data file names.  
or S and EXEC for Scratched Programs and Data file names.  
or ALL and EXEC for sequenced list of AP, AD, SP, and SD file names.  
or DATA and EXEC for sequenced list of AP, AD, SP, and SD file names contained within a list of DATA statements overlaid over lines 9000+. The output list will show files contained on the surface and also denote those missing.

Prompt 5 Output to: 0=CRT 1=215 2=204 3=216 (DEFAULT)= 0 ?

Respond EXEC or digit and EXEC.

Prompt 6 List wanted 0=Cat.data 1=plus REMS (DEFAULT)= 1 ?

Note "plus REMS" means to display a catalog index line and:  
for programs a portion of the first line of a program file if coded as a REM or % (image statement).  
for data files a description for the data file name if found in a list of DATA statements overlaid over lines 9000+.

Respond EXEC or digit and EXEC.

Prompt 7 INPUT NON-ZERO FOR MORE DATA? \_  
Respond EXEC to start Sort collection \_  
or key and EXEC to prompt for an additional sort collection..  
-----

Additional options:  
These options may be invoked after the collection phase of the sort.

'1-Cat.only --  
displays for each file only the single line of catalog information.

'2-Cat.with REMs --  
displays for each file the catalog line plus a REM. (see prompt 6 elaboration).

'7=BEGIN --  
starts the display over again with the first item.

'14 date + --  
prompts for a date mm/dd/yy and then searches for program files entered that date or after. The program must have an initial REM or % statement with a date included. The included date may be of mm/dd/yy or yy/mm/dd format.

'15 RECALL ID  
Skip through the list of collected names and begin with a matching ID.  
-----

Output:

Screen DISK CATALOGUE SORTED BY NAME  
INDEX SECTORS = nn  
END CAT. AREA = aaaaa  
CURRENT END = aaaaa  
SEARCHING FOR AP [Standard index/New index method  
CAT. SECTOR= s FOUND ITEM = nn filename  
AP ITEMS FOUND= ccc

ITEM	NAME	TYPE	START	END	USED	FREE	+USED
1	TBO DIAG	P	S.s.#	E.s.#	6	1	6
	text of REM or % statement if first line of program.						
2	TBO UTIL	P	S.s.#	E.s.#	6	2	12
	text of REM or % statement if first line of program.						

'1-Cat.only '2-Cat.with REMs '7=BEGIN '14 date + '15 RECALL ID

Printer DISK CATALOGUE SORTED BY NAME  
INDEX SECTORS = nn  
END CAT. AREA = aaaaa  
CURRENT END = aaaaa  
AP ITEMS FOUND= ccc

ITEM	NAME	TYPE	START	END	USED	FREE	+USED
1	TBO DIAG	P	S.s.#	E.s.#	6	1	6
	text of REM or % statement if first line of program.						
2	TBO UTIL	P	S.s.#	E.s.#	6	2	12
	text of REM or % statement if first line of program.						

-----



**Abstract:** This program displays a clock and calendar on a 2200 terminal. The clock and calendar can be used for reminder messages for the day. If a universal global area is set reminders may be sent from terminal to terminal.

**Modules:** @CLOCK -- DATETIME -- TBO.MSGS

**Equipment used:** CRT / keyboard and MXE controller.

**Operating instructions:**

for "DATETIME"

Display Enter Date and Time  
MM/DD/YY HH:MM

Respond Valid date and time for the MXE controller.

for "@CLOCK" Display

FRIDAY JANUARY 24, 1986

W A N G 2 2 0 0 M i c r o - V P

```

***          *****          *****          *****          *****          *****
**          **   **          **   **          **   **          **   **          **   **
**          **   +   **   **          **   **          **   **          **   **
**          **          **   **          **   **          **   **          **   **
**          **   +   **   **          **   **          **   **          **   **
**          **          **   **          **   **          **   **          **   **
*****      *****          *****          *****          *****          *****

```

JANUARY 1986

SUN	MON	TUE	WED	THU	FRI	SAT
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	<u>24</u>	25
26	27	28	29	20	31	

FEBRUARY 1986

SUN	MON	TUE	WED	THU	FRI	SAT
						1
	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

Respond space to find instructions  
 Key NOTE (s.f. '6) to display messages for this terminal  
 -- permanent reminders are DATA statements at lines 6100+  
 Key INSERT to enter current reminder messages  
 Key DELETE to remove messages  
 Key <- PREV or NEXT -> to shift months display  
 Key CANCEL/EDIT to return to main menu  
 Key EXEC to show current month.

Respond CANCEL/EDIT to exit to main module saving outstanding reminders.

NOTE: Reminder messages to display in BOLD TYPE and standard type may be inserted to occur at set times. Nine characters are displayed BOLD TYPE.

NOTE: Holidays and weekends are displayed in highlighted reverse intensity. The current day is displayed in boxed in high-intensity blinking.

Holidays are coded in A\$="mm/dd, ... " form on line 125.

Additional instructions for @CLOCK.

hh:mm START.JOB FileName —Special NOTE form to LOAD RUN "FileName"  
Key '31 to send a message to another terminal.

respond '10 to INSERT messages to your terminal's Clock file.  
display

hh:mm BOLD TYPE insert message  
hh:mm text for message 1  
hh:mm text for message 2

respond hh:mm text for new message  
or mm/dd hh:mm text for new message

respond '9 to DELETE messages from your terminal's Clock file.  
display

hh:mm or ALL -- DELETE message  
hh:mm text for message 1  
hh:mm text for message 2

respond hh:mm and RETURN to delete one of today's messages  
or mm/dd hh:mm and RETURN.  
or ALL and RETURN

## Enhancements for an improved clock functionality.

1. File "SYS CRTS" may be added to describe terminals on your system.  
NOTE descriptions for system terminals are in a file "SYS CRTS"

```
8000 REM .SYS CRTS
      :REM. "Msg.File # CRT description
8010 DATA "@CLOCKM" 1 Main system terminal
8020 ... "MSG.S.TBO 2 Main system terminal
```

2. A global message area may be established for inter-terminal communications.  
Line 24 of @CLOCK looks for this partition.

```
0024 MO=1
      : SELECT @PART "3270UNIV"
      : ERROR MO=0
```

3. Use a file called "SYS COM" as the first Universal COMMON memory module.

```
0010 REM SYS COM mm/dd/yy COMMON memory for this system
0015 REM .Called from WAITDATE
0020 COM @M$(8)50
      : REM /.# Terminals for CLOCK messages.
0070 LOAD T "3270U2"          Calls next Universal partition module.
```

Function: What disks are on the system?

Modules: TBO.XDAD

Abstract: This program displays the status of all disks on the system. It is a static display which goes through possible disk addresses on 10, 20, and 30

It shows Cur.end -Max., Access errors, disk not configured, disk unavailable.

What disks are attached to the system

Disk controller 10 Disk type D0

1 310 Yes Cur.END=17414 Max.=38911 ! 2 B10 ERR.I98 Platter not mounted !  
 2 D11 Yes Cur.END=17414 Max.=38911 ! 4 D10 ERR.I98 Platter not mounted !  
 5 D12 ERR.I91 None

Disk controller 20 Disk type C0

1 320 Yes Cur.END= 1231 Max.= 1231 ! 2 B20 Access error = 98 !.  
 3 360 ERR.I98 Platter not mounted !

1 330 Yes Cur.END=17650 Max.=65023 ! 2 B30 etc. !.  
 3 370 Yes ..etc  
 5 D31 Yes ..etc

THIS IS THE STATUS OF POSSIBLE DISK ADDRESSING ON THIS SYSTEM

Key '0 to EXIT '1 to TBO.STAT '15 RECALL - other for disk notes

(canned informational display)

Shugart -- Dual or triple (1231) maximum sectors, white label  
 Winchester (3873) maximum sectors, red label  
 2275's -- DSDD max sectors (1292) maximum sectors, 5 1/4 inch  
 2275-30 -- DSDD on D.0 (1292) Winchester on D.1 and D.2 (18900)  
 2275-60 -- No diskette -- 4 Winchester surfaces on D.0 thru D.3 (65023)  
 Phoenix -- No diskette -- surfaces on D.0 thru D.F (52763)

Function: Show status of Printers, T.C. boards, and Terminal

Abstract: This program displays the status of peripherals on selected device addresses. Printers are tested on 215, 216, 217, and 218. T.C. boards are tested on 6 addresses from 1A to 1F. The status of the operating system and user terminal are also displayed.

Modules: TBO.STAT

Screen display

2200 Device status

Terminal status

!\*2436DE R0101 19200B 8+0 (USA) !

Terminal number	1	! CPU type	MVP	!
Partition number	9	! MVP Release	2.6	!
Partition memory	56	! CRT size	80	!

Printers

215 Unavailable	! 216 Not Ready	!
217 Unavailable	! 218 Unavailable	!

T.C. Boards

01A Unavailable	! 01B Unavailable	!
01C B. 32K	! 01D D. 64K	!
01E B. 8K	! 01F Unavailable	!

Key '0 to START '1 to What disks? other key to retest here

Function: LOAD RUN xx from another surface

Abstract:

The program does a SELECT #1 with the given disk address. It then tries a LIMITS T #1, with the file name given to assure the file name given is a valid program on the stated disk surface.

If false, reprompting occurs. S.F. '0 may be used to reload "START".

If true, TBO.LRUN does a SELECT #0 with the new disk surface; the new program is activated with a LOAD RUN command.

Modules: TBO.LRUN

Display Activate a system from another platter

Prompt 1 On disk surface \_\_\_\_

Respond hhh and EXEC where hhh is a valid disk address.

Prompt 2 Program START

Respond EXEC to load "START".

else file name and EXEC.



**Function:** Cross-reference listing

**Abstract:** One or more program files on any system disk may be listed on numbered and titled pages. For each program requested the listing consists of decompressed program statements followed by a cross-referenced listing of variables used, special function, and numbered statement references.

**Modules:** TBO.CRF0 - TBO.CRF1 - TBO.CRF2 - TBO.CRF3

**Equipment used:** CRT / keyboard, disk and optional printer.

**Operating instructions:**

**Display** TBO xref -- System code = MVP with 56 K memory  
 CROSSRF -- S.F. Entry points  
 From prompt module:  
 '0 Change Disk media  
 '1 Pick-up prompting in program names  
 '5 display verb atoms '15 Show S.F. actions  
 '12 writeup '14 List options  
 From cross-reference listing mainline:  
 '10 Summary to Printer/CONTINUE '11 Summary only

**Prompt 1** Output to: 0=CRT 1=215 2=204 3=216 (DEFAULT)= 0 ? \_  
 Respond EXEC or digit and EXEC.

**Prompt 2** DATE?  
 Respond any character stream (no commas) will be part of a large print title put at the top of each printed page.  
 EXEC or digit and EXEC.

**Prompt 3** 1=SAVE MEANINGS THRU ALL PROGRAMS?  
 Respond EXEC if you are not sure.  
 or 1 and EXEC if variable meanings are described and are to be carried from program to program.

**Prompt 4** For annotation KEY -- 0=Summary only 1=In Listing 2=In Margin  
 DEFAULT=0 ?  
 Respond EXEC if you are not sure.  
 or 1 and EXEC to get meanings embedded in the listing.  
 or 2 and EXEC to get meanings placed in the right margin of the listing.

**Prompt 5** PAPER WIDTH NARROW=0 WIDE=1 DEFAULT=0 ?  
 Respond EXEC or 0 and EXEC for 80 column listings  
 or 1 and EXEC for 120 column listings

**Prompt 6** List mode KEY--0=List + XRef 1=List only 2=XRef only  
 DEFAULT=0 ?  
 Respond EXEC or 0 and EXEC to get a complete listing.  
 or 1 and EXEC to get a listing of only the program text,  
 or 2 and EXEC to get a listing of only the cross-reference summary.

**Prompt 7** START STMT DEFAULT=0 ?  
 Respond EXEC to get a listing of all lines of the program.  
 or 1 and dddd EXEC to get a listing beginning at a specified line.

Prompts 8 and 9 are repetitive through a list of file names.

Prompts 8 and 9 are repetitive through a list of file names.

Display 1 =310 2 =B10 3 =320 4 =B20  
 Prompt 8 Disk surface Key single digit or any valid disk address 1  
 Respond EXEC  
 or digit and EXEC  
 or disk address and EXEC  
 Display 1 =310

Prompt 9 Disk Program Name + commentary. S.F.'1 ON ERROR 80)? \_\_\_\_\_  
 Respond Program name (8 characters) plus commentary to append to title.  
 and EXEC prompt 9 will repeat until a single EXEC is entered.  
 or DATA and EXEC to get a list of program names from a list of DATA  
 statements which may loaded to overlay the code at 9000+.  
 or ALL and EXEC to localize the search by a common 0-8 byte IDentity.  
 or ALL- and EXEC to exclude modules with a common 1-8 byte IDentity.  
 or '0 to select an alternative disk surface.  
 or EXEC to terminate the list of items.

NOTE: once the prompting sequence has gotten to Prompt 8 or Prompt 9 the  
 S.F. keys are very useful to pick-up within the sequences.  
 S.F. ' 0 may be used to select a new disk surface,  
 and S.F. '1 to pick-up in the list of file names.

-----

Output:

Section 1 - An index page to the program files listed, disk surface and  
 description.

Program listing:

Titled and page numbered listings of program files.  
 Section of decompressed program code.

Summary cross-reference section  
 Variables

Special functions

Statement numbers.

-----

The cross-reference program will create a listing of any BASIC  
 program file. The listing will consist of titled and numbered pages. Each  
 listing contains a program listing where each segment of a multi-statement  
 line appears on a separate printer line. The listing will contain a blank  
 line after each GOTO, RETURN, LOAD, or STOP statement. The listing will be  
 automatically indented following IF statements, if not followed by another IF  
 statement, and following FOR statements.

A separate summary section lists:

- 1) # references and variables used, meanings (optional), and  
 referencing statement lines.
- 2). Special functions and meanings (optional).
- 3). Statement number and special function cross-references.

**SPECIAL OPTION -- PAGE FORMATTING:**

The CROSSRF program is written to interpret certain REM statements in a way that will enhance program documentation. REM% statements may be incorporated into the source BASIC causing special listing effects.

REM% - REM (per cent) (up arrow) (comment)  
positions to a new printer page with title and expanded print comment.

REM%d- REM (per cent) (digit 1-9) (comment)  
skips n lines and prints the comment in expanded type.

REM% comment - REM (per cent) (comment)  
skips 2 lines and prints the comment in expanded type.

:REM/ comment - (colon) REM (slash) (comment)  
puts the comment in the right column of the previous statement.

**SPECIAL OPTION -- ENTRY OF MEANINGS TO VARIABLES AND SPECIAL FUNCTIONS:**

A descriptive meaning may be assigned to each variable, # file reference, or special function. This meaning, if used will be automatically output in the cross-reference summary listing. The meaning may also be output in the body of the listing either embedded into the program code or in the right margin on each occurrence of the variable or special function. The descriptive material is entered into special tables by special REM statements. The special statements, described in the below, may be entered either in the program text or in a separate program file. Any variable may be given a new meaning merely by entering a new meaning item. Refer to question 3 in the interactive sequence for activating the meanings option.

REM%0 -REM (per cent) (zero) causes an entry to be placed into the meanings table. Meanings entries are coded as follows:

	name space description comma
or	name space description colon
or	name space description carriage return
where	name is up to 4 characters with no spaces, eg A0\$( space is one or more spaces, description is 1 to 16 characters of description (no commas) comma, colon, or carriage return terminates the meaning.

Example of BASIC program code required to enter meanings.

```
30 REM%0 A A, B CRT line, C Start char pos, C$ C$1, L Field size
40 REM%0 '201 Edit input, #5 CRT
```

**TABLE CAPACITIES**

The CROSSRF program does not use any storage other than CPU memory. In order to create cross-reference summaries it is necessary to store certain table in system memory. Memory space for these tables is coded into COM statements in the loader module (TBO.CRFO). A separate set of COM statements for each memory size has been incorporated on the program disk. The default values for COM statements of various memory configurations may be modified if necessary.

**Function:** Program or system comparison

**Modules:** TBO.CMPS

**Equipment used:** CRT / keyboard, disk and optional printer.

**Abstract:** Program files on any system disks are compared line by line. Differences between the two files are displayed line by line. Two system disks may be compared by activating a list of DATA statement names.

The program comparison utility operates in one of two modes:

- 1). program mode allows the comparison of single program files. The programs may have the same name on different disk surfaces or different names on the same disk surface.
- 2). system mode allows the comparison of a list of program files occurring on two different disk surfaces. The list of names is from DATA statements overlaid over lines 9000 and beyond. The operator keys DATA when prompted for the first input file name.

Output to the CRT is scrolled and will show:

#1 IS ON LINE = 0010

#2 IS ON LINE = 0010

#1 IS ON LINE = 0020

#2 IS ON LINE = 0020

#1 --

0020 COM @Z\$(26)80, @X\$(2)2, ... text for line with differences.

#2 --

0020 COM @Z\$(26)80, @X\$(1)2, ... text for line with differences.

If output is to the CRT only, the program will pause at each differing line. The operator may key any key to continue the search. Keying RUN will allow the program to continue the search of the current files without pause.

If output is to the printer the program will continue without pause through the entire list of programs to be compared.

Operating instructions:

Display PROGRAM COMPARISON

NOTE— TO COMPARE SYSTEMS WITH INPUT VIA DATA STATEMENTS

ENTER THE NAME DATA AS THE 1ST PROGRAM NAME FOR #1

Key S.F. '1 to pickup in entry of program names

prompt KEY 0 if output to CRT?

respond 0 and RETURN for output to the CRT

or 1 RETURN for output to Printer

additional prompts for printer output only

prompt # OF COPIES?

prompt COMMENT ?

respond comments for up to 9 lines of commentary for the top of the listing.

or space and RETURN to end the COMMENTS.

display FIRST PROGRAM---#1

1 =310 2 =B10 3 =320 4 =B20

Prompt Disk surface Key single digit or any valid disk address 1\_\_

Respond EXEC

or digit and EXEC

or disk address and EXEC

Display 1 =310

prompt PROGRAM #1 NAME? \_\_\_\_\_

respond 8 character program name and EXEC

or DATA and EXEC

display SECOND PROGRAM---#2

1 =310 2 =B10 3 =320 4 =B20

Prompt Disk surface Key single digit or any valid disk address 1\_\_

Respond EXEC

or digit and EXEC

or disk address and EXEC

Display 1 =310

the prompt below will occur if DATA was not entered for PROGRAM #1.

prompt PROGRAM #2 NAME? \_\_\_\_\_

respond 8 character program name and EXEC

or DATA and EXEC



**Function:** Analyze data file structure

**Modules:** TBO.ANDF

**Abstract:** This program analyzes a catalogued data file for structure. The output is a summary of the SAVE statements used to create the file.

**Example** File name=@SYSFILE  
File size = 32 Sectors.  
File=@SYSFILE BASIC-2 data structure -- SAVE number 1  
\$8 Scalar \$32 \$( 2)16 \$(16)8 \$(33)3 \$(15)13

**Example** File name=BSC\*010A  
File size = 40 Sectors.  
File=BSC\*010A BASIC-2 data structure -- SAVE number 1  
\$( 256)16  
File=BSC\*010A BASIC-2 data structure -- SAVE number 2  
\$( 256)16  
END OF FILE

Operating instructions:

Display A n a l y z e D a t a F i l e F o r m a t

\$(4)62 Wang T.C. file format  
\$2, \$3, \$(4)60 Wang Prom file format  
\$(256)16 4K TC microcode  
\$(512)16 8K TC microcode  
\$(34)121 4K TC ucode (min.disk storage)  
\$(3)83 3270 Spooler names file  
\$(3)80 3270\*PQ  
\$31, \$(128)2, Scalar(3) VFU format tape  
\$8, Scalar, \$32, \$2(16), \$(16)8, \$(33)2 @SYSFILE

Key '15 to see file examples  
1 =310 2 =B10 3 =320 4 =B20

Prompt Disk surface Key single digit or any valid disk address 1

Respond EXEC  
or digit and EXEC  
or disk address and EXEC

Display 1 =310

display see example above.



(K) Prompts for File Z A P

Description of possible operations.

prompt '0 Change sector

respond dddd and RETURN valid disk sector address.

of

prompt '1 Change device

respond hhh and RETURN valid 3 hexdigit disk address.

prompt '3 HEX to decimal

respond hh and RETURN valid hex values.

'4 Exit program

'9 Find ASCII string

respond text and RETURN valid ASCII data to find.

'10 Find HEX string

respond text and RETURN valid HEX data to find.

'11 Find start of file

respond valid file name and RETURN.

'12 Next sector

'13 Previous sector

'14 Print sector

1-2183

1-2183

1-2183

1-2183

1-2183

.ben

1-2183

1-2183



**Abstract:** From the "START" module this function analyzes disk surfaces to find and correlate all program and data file references. Program modules found are of two types: menu modules are a list of DATA statements originated at BASIC-2 line numbers 9000 and above. Function modules are all other program modules called either from menus or other programs. The program opens in two parts: Part one "TBO.DMAP" goes through the menu structure and creates two common tables listing menu modules and non-menu program modules. Part two "TBO.FMAP" goes through all the program function modules to find all program LOAD and LOAD at DATA LOAD OPEN statements.

**Equipment used:** CRT / keyboard and optional printer.

**Operating instructions:**

**Display**      **Function Analyzer**  
Source      Source Program disk -  
1      1 =310      2 =B10      3 =320      4 =B20  
Prompt 1      Disk Surface - Key single digit or any valid disk address 1  
Respond      EXEC  
or      Digit and EXEC  
or      Disk address and EXEC  
Display      1 =310  
List of programs is from COMMON P\$() array  
1 START  
2 @CLOCK

**Respond**      PROGRAM NAME?  
or      EXEC  
or      program name and EXEC  
KEY #1 (EXEC) FOR HARD COPY?

**Display**      (High-lighted Program name)      ..SELECTED LINES Prog. 1 of 166

(listing of all LOAD statements found in program)

(High-lighted Program name)      VERB=LOAD IS USED d TIMES  
referenced

P Menu      0001 STARTTBO  
PROGRAM LOAD      PROGRAM LOAD is at 0010  
LOAD is at      LOAD is at --0010

**Example**      **Example**      STRT3275      STRT3275 ..SELECTED LINES Prog. 28 of 166

0040 DATA 10040 DATA LOAD DC OPEN T #0, "M-3270"  
0060 DATA 10060 DATA LOAD DC #0, W\$()  
0150 LOAD 01150 LOAD DC T "EM3275"

STRT3275      STRT3275 VERB=LOAD VERB=LOAD IS USED 3 TIMES

Files referenced  
D      M-3270      M-3270  
Function      EM-3275  
PROGRAM LOAD      PROGRAM LOAD is at 0150  
DATA LOAD      DATA LOAD OPENS at 0040  
LOAD is at      0040-0060-0150

EVERCPUB, EVERCPUC, EVERCPU

1/21/94

1/21/94 IF SELECT SCSI PICK FROM BACKUP, RESTORE, OR THE DISK MANAGEMENT WINDOW, RETURNS YOU BACK TO THE MENU WITH A MESSAGE.

FIX: ADDED LINE 25 TO ALL 3 PROGRAMS

25 IF STR(A\$,9,1) = "T" THEN 40: PRINT AT(3,0,50); HEX(0E); "DEVICE NOT SUPPORTED!!"; HEX(0F 01): R6M - NON-TURBO CPU



# B-TAGG (IF/THEN, ELSE TEST)

8/18/94

GBT ERROR EVERY PASS & NEVER LEAVES THIS SCREEN OF TESTS.

8/19/94 ELSE STATEMENT IGNORED ON CS/2200 O/S 2.1 & BELOW IF NOT ON SAME LINE AS IF/THEN.

LINE 730 INCORRECTLY TESTS  $X \lt \> \emptyset$ . SHOULD BE TESTING  $X \lt \> 2$  AS IF WORKING CORRECTLY X WILL ALWAYS BE EQUAL TO 2.

**FIX** 700  $X = \emptyset$ : IF  $1 < 2$  THEN  $X = 1$ : ELSE  $X = 2$

710 IF  $X \lt \> 1$  THEN 8990

720  $X = \emptyset$ : IF  $2 < 1$  THEN  $X = 1$ : ELSE  $X = 2$

730 IF  $X \lt \> 2$  THEN 8990

2/26/99

CHANGES FOR MA TRUCK BODY IN CHELSEA, MA FOR YEAR 2000

MBNKKFAM - USED TO SET SYSTEM DATE ON BOOT. MADE COPY - MBNKKFA1

~~MBNKKFAM~~ LINE 3080 ~~DELETED : IF U9 = Ø THEN 3100 AT END~~

MEMNB/UP - CUSTOMIZED COPY OF CBACKUP FOR TRAILER EQUIPMENT SALES

~~MEMNB/UP~~ LINE 900 ~~DELETED : IF U9 = Ø THEN 920 AT END~~

900 REM CHECK FORM: IF VER(U9\$, "##/##/##") <> 8 THEN 920:

REM CHECK MO.: CONVERT STR(U9\$, 1, 2) TO UØ: IF UØ < 1 OR UØ > 12

THEN 920: REM CHECK YR.: CONVERT STR(U9\$, 7, 2) TO U9

~~CBACKUP-5/13/98~~ VER ~~DELETED : IF U9 = Ø THEN 920 AT END OF LINE 900~~

TO ALLOW USE OF YEAR ØØ

2/26/99

## CHANGES FOR MA TRUCK BODY IN CHELSEA, MA FOR YEAR 2000

MBNKKFAM - USED TO SET SYSTEM DATE ON BOOT. MADE COPY - MBNKKFAM

MBNKKFAM LINE 3080 DELETED: IF U9 =  $\emptyset$  THEN 3100" AT END

TEMNB/UP - CUSTOMIZED COPY OF CBACKUP FOR TRAILER EQUIPMENT SALES

TEMNB/UP LINE 900 DELETED: IF U9 =  $\emptyset$  THEN 920" AT END

NOW 900 REM CHECK FORM: IF VER(U9\$, "##/##/##") < > 8 THEN 920:

REM CHECK MO.: CONVERT STR(U9\$, 1, 2) TO U $\emptyset$ : IF U $\emptyset$  < 1 OR U $\emptyset$  > 12

THEN 920: REM CHECK YR.: CONVERT STR(U9\$, 7, 2) TO U9

CBACKUP-5/13/98) VER DELETED: IF U9 =  $\emptyset$  THEN 920" AT END OF LWB 900

TO ALLOW USE OF YEAR  $\emptyset\emptyset$

## INSTALLATION INSTRUCTIONS:

AFTER SYSTEM IS UP & TERMINAL 1 IS AT MAIN MENU # 1:

1. INSERT FLOPPY DISK IN FLOPPY DRIVE
2. SELECT '16 - DISK UTILITIES FROM MAIN MENU
3. USE SPACE BAR TO SELECT 'MOVE FILE' & KEY RUN
4. CHANGE "INPUT ADDRESS" TO D10 TO READ FROM FLOPPY
5. KEY RETURN TWICE, TO ACCEPT D10 ADDR & WANG PLATTER TYPE
6. CHANGE "OUTPUT ADDRESS" TO D11
7. KEY RETURN TWICE, TO ACCEPT D11 ADDR & WANG PLATTER TYPE
8. ENTER Y WHEN ASKS IF WISH TO MOVE ALL ACTIVE FILES
9. KEY RETURN TWICE, TO ACCEPT CHANGE & TO OVERWRITE FILES.
10. 3 FILES WILL BE UPDATED. KEY CANCEL/EDIT TO RETURN TO MENU WHEN DONE

11/15/96

CHANGES FOR MA TRUCK BODY IN CHELSEA, MA TO ADD ACCESS TO DISK UTILITIES MENU FROM THEIR MAIN MENU & TO INCREASE PARTITION SIZE TO ALLOW DSTAPEB, TAPE BACKUP, TO RUN.

From DISK UTILITIES 1.10.00, THE FOLLOWING FILES WERE TAKEN W/O CHANGE:

.STARTD	CDSTAPER	CTOIMAGE
CDSAPPLY	CHITRATE	CTO.CRF
CDSCFIG	CMOVE1	CTO.CRFØ
CDSCFIGP	CRAMPDISK	CTO.DISK
CDSTAPEB		CTO.SUBS

From DISK UTILITIES 1.10.00, THE FOLLOWING FILE WERE TAKEN & EDITED AS SHOWN:

1. START LINE 10 MADE \$PSTAT = " " INSTEAD OF "SYSMVPA"  
RENAMED PROGRAM TO DISKUTIL.

CHANGE TO \$PSTAT ALLOWS DEFAULT TO ROOT NODE & CAN PREVENT PROBLEMS CAUSED BY COMMON VARIABLES. HAD TO RENAME PROGRAM AS START ALREADY EXISTED.

2. CMENU LINE 120 AFTER 1<sup>ST</sup> : ADDED IF UB = "TEMNMENU" OR UB = "YESI"  
THEN 125:

ADDED NEW LINE 125 LOAD RUN#U2, "TEMNMENU"

CUSTOMER NOT USING CMENU FOR THEIR MAIN MENU, WHEN KEYING CANCEL/EDIT TO RETURN TO CUSTOMER'S MENU, TEMNMENU, WOULD GET PS6 ON LINE 200.

TEMNMENU WAS OVERLAYING CMENU CAUSING A MIX OF BOTH PROGRAMS.

3. CMOVEFIL LINE 550 CHANGED LOAD RUN "START" TO LOAD RUN "DISKUTIL"

CHANGE REFLECTS RENAMING OF START TO DISKUTIL. LINE IS USED WHEN KEY FN/TAB TO RETURN TO DISK UTILITIES MENU.

4. CSYSMVPA LINE 9020 CHANGED "SYSMVPA" AT END TO "TEMNMENU"

THIS IS THE DISK UTILITIES MENU & THIS CHANGE ALLOWS YOU TO CANCEL/EDIT BACK TO THE CUSTOMER'S MENU, TEMNMENU, INSTEAD OF LOOKING FOR SYSMVPA.



5. C.BACKUP LINE 9040 REM'D LINE

ELIMINATES SCSI TAPE BACKUP FROM BACKUP UTILITIES MENU.

6. C.DISK LINE 9050 REM'D LINE

ELIMINATES SCSI CONFIGURATION FROM DISK MANAGEMENT MENU.

7. C.RESTOR LINE 9040 REM'D LINE

ELIMINATES RECOVER FROM SCSI TAPE FROM RESTORE UTILITIES MENU.

THE FOLLOWING CUSTOMER FILE WAS EDITED:

TEMNMENU ADDED NEW LINE 4115 PRINT TAB(43); "16 - DISK UTILITIES"

LINE 4150 ADDED ,4325 BEFORE: GOTO 4130 AT END

ADDED NEW LINE 4325 GOSUB '33("DISK UTILITIES"): COM CLEAR:

LOAD DCT #0, "DISKUTIL"

ADDS DISK UTILITIES PICK TO CUSTOMER'S MAIN MENU.

THE DS TAPE BACKUP PROGRAM, CDSTAPER, WOULD NOT RUN IN A 23.5K PARTITION THE CUSTOMER WAS USING. CREATED NEW CONFIGURATION FILE, MATRUCK, IN CGENPART WITH LARGER PARTITION SIZES TO TAKE ADVANTAGE OF ADDED MEMORY. OLD CONFIGURATION FILE, "TESI", STILL EXISTS WITH CUSTOMER'S ORIGINAL CONFIGURATION. NEW CONFIGURATIONS:

PARTITION	MEMORY	TERMINAL	PROGRAMMABLE	PROGRAM TO LOAD
1	3.0	1	Y	KFAM0407
2	47.0	1	Y	TESI
3	11.0	1	Y	MBNKKFAM
4	45.0	2	Y	TEMNMENU
5	11.0	2	Y	MBNKKFAM
6	56.0	3	Y	

RELATED FILES INCLUDED: CGENPART (LATEST VERSION), CSYSFILE (CONFIGURATION FILE)



TO INSTALL CHANGES TO ADD BACKUP UTILITIES TO SYSTEM:

MAKE SURE HAVE CURRENT BACKUP BEFORE STARTING.

1. MAKE SURE NO ONE IS USING SYSTEM AND ALL JOBS ARE COMPLETED.
  2. FROM MAIN MENU #1, KEY SHIFT/RESET. SCREEN IS BLANK EXCEPT FOR "READY (BASIC-2) PARTITION 02" AT TOP.
  3. TYPE IN CLEAR THEN HIT RETURN KEY
  4. TYPE IN SELECT DISK D10 THEN HIT RETURN KEY
  5. INSERT DISKETTE WITH CHANGES IN FLOPPY DRIVE & CLOSE LATCH.
  6. TYPE IN LOAD RUN "DISKUTIL" THEN HIT RETURN KEY  
DISK UTILITIES 1.1 MENU SHOULD DISPLAY ON SCREEN.
  7. USE SPACE BAR KEY TO MOVE ACCEPTANCE BLOCK TO 'MOVE FILE' PICK.  
HIT RUN KEY TO RUN MOVE FILE PROGRAM.
  8. SCREEN SHOULD NOW BE PROMPTING YOU FOR "INPUT ADDRESS: D11"  
CHANGE THIS ADDRESS TO D10 THEN KEY RETURN.
  - \* IF AT ANY OF THE FIELDS IN THE 'MOVE FILE' UTILITY YOU MAKE A MISTAKE & NEED TO GO BACK TO CORRECT, HIT THE FN &/OR TAB KEY & RESTART AT STEP 7.
  9. NOW ASKS FOR "INPUT PLATTER TYPE: W". HIT RETURN TO LEAVE AS W.
  10. CHANGE OUTPUT ADDRESS TO D11 & HIT RETURN KEY.
  11. HIT RETURN TO LEAVE "OUTPUT PLATTER" AS TYPE W.
  12. MAKE SURE INPUT ADDRESS = D10, OUTPUT ADDRESS = D11 & BOTH ARE TYPE WANG. KEY FN/TAB KEY & RESTART AT STEP 7 IF CORRECT.
- SYSTEM IS NOW ASKING IF YOU WISH TO MOVE ALL FILES. CHANGE THE N TO Y & KEY RETURN.
13. NOW ASKS IF YOU WANT TO OVERWRITE. LEAVE AS Y & KEY RETURN.  
SYSTEM WILL NOW BEGIN MOVING FILES TO YOUR SYSTEM DISK. WHEN DONE (ABOUT 5 MIN), SYSTEM WILL RETURN TO DISK UTILITIES SCREEN

14. SYSTEM MUST NOW BE REBOOTED TO UTILIZE CHANGES MADE.  
KEY SHIFT/RESET TO CLEAR SCREEN.
15. TYPE IN "BINIT" SYSTEM" THEN RETURN TO REBOOT  
SHOULD COME BACK W/ MOUNT SYSTEM PLATTER  
PRESS RESET
16. REBOOT SYSTEM AS YOU NORMALLY WOULD FROM THIS SCREEN.

HIT YOUR MAIN MENU #1 YOU WILL NOW SEE AN ADDITION PICK,  
'16 - DISK UTILITIES

USE SF'16 (SHIFT/SF'16) TO ACCESS THE DISK UTILITIES FROM  
WHICH YOU CAN RUN YOUR BACKUP.

TO RUN BACKUP FROM MAIN MENU #1,

1. KEY SF'16. DISK UTILITIES MENU IS DISPLAYED.
2. USE SPACE BAR TO MOVE ACCEPTANCE BLOCK TO BACKUP UTILITIES &  
KEY RUN.
3. BACKUP DISK PLATTERS TO DT TAPE CASSETTE SHOULD BE SELECTED. KEY RUN.
4. SCREEN PROMPTS FOR TAPE ADDRESS. DSF IS CORRECT. HIT RETURN.
5. APPEND OR ERASE. APPEND ALLOWS YOU TO ADD THIS BACKUP TO CURRENT  
... END OF TAPE ALLOWING PREVIOUS DATA OR BACKUPS TO REMAIN.  
ERASE CLEARS TAPE. COULD PROBABLY FIT 4 OR 5 BACKUPS OF  
DII ON 1 TAPE BEFORE RUNNING OUT OF SPACE. CHOOSE A OR E & KEY  
RETURN.
6. NOW ASKS FOR ADDRESS OF DISK TO BACKUP. TYPE IN D11 & RETURN.
7. ENTER LABEL FOR BACKUP MINIMALLY TODAY'S DATE & KEY RETURN.
- 8... START SECTOR MUST BE 0. KEY RETURN.
9. ACCEPT DEFAULT END SECTOR BY KEYING RETURN.
10. KEY RETURN TO ACCEPT ENTRY. IF NEEDED TO CORRECT HIT N RETURN & GO TO 6
- 12... IF D11 IS ONLY BACKUP BLANK OUT ADDRESS & KEY RETURN, OTHERWISE REPEAT  
STEPS 6 THROUGH 10 FOR EACH ADDRESS TO BACKUP.

13... WHEN BACKUP COMPLETES, KEY FN/TAB KEY. RETURNS SYSTEM  
... TO BACKUP MENU.

14... KEY CANCEL/EDIT TO RETURN TO DISK UTILITIES SCREEN.

15... KEY CANCEL/EDIT TO RETURN TO MAIN MENU #1.