```
**********************************************************
*                                                        *
*                     B 2 2 0 S I M                      *
*                                                        *
*                Burroughs 220 Simulator                 *
*                                                        *
*   Written by Michael J. Mahon   -   March 21, 2016     *
*                                                        *
* The B220 is a BCD word-oriented computer with 5000     *
* 11-digit words in the following format:                *
*      ___________________________________________       *
*     | S | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 |      *
*     |___|___|___|___|___|___|___|___|___|___|___|      *
*                                                        *
* If the sign digit (S) is even, the number is positive, *
* if odd, negative.  If S is 2, the word is interpreted  *
* as five alphanumeric characters.                       *
*                                                        *
* "Partial fields" may be specified within a word by a   *
* 2-digit partial field specification, sL, where s is    *
* the rightmost digit of the field and L is the length,  *
* extending to the left no further than the Sign digit.  *
*                                                        *
* Decimal floating-point data is stored in this format:  *
*                                                        *
*      ___________________________________________       *
*     | S | E | E | M | M | M | M | M | M | M | M |      *
*     |___|___|___|___|___|___|___|___|___|___|___|      *
*                                                        *
* S is the sign of the mantissa, as for fixed-point data.*
*                                                        *
* EE is the excess-50 power of ten.                      *
*                                                        *
* MMMMMMMM is the fractional, normalized mantissa.       *
*                                                        *
* Instructions have the following format:                *
*                                                        *
*      ___________________________________________       *
*     | S | V | V | V | V | O | P | A | D | D | R |      *
*     |___|___|___|___|___|___|___|___|___|___|___|      *
*                                                        *
* If S is odd, ADDR is modified by the B register before *
* use.                                                   *
*                                                        *
* The Variant field (VVVV) has an op-specific format.    *
*                                                        *
* The OP field is the opcode.                            *
*                                                        *
* The ADDR field is the address part of the instruction  *
* which is augmented by B if the Sign digit is odd.      *
*                                                        *
**********************************************************
```

```

         put   B220HISTORY
**********************************************************
*                                                        *
*                       History                          *
*                                                        *
* 03/29/16 - Ran first B220 op--HLT!  BCD address to MEM *
*            address is OK.                              *
*                                                        *
* 03/31/16 - Began implementing B220 front panel display *
*            in 40-column text mode.                     *
*                                                        *
* 04/02/16 - Front panel complete, adding keyboard cntl. *
*                                                        *
* 04/05/16 - Keyboard control complete, adding opcodes.  *
*                                                        *
* 04/11/16 - Refined error handling. Added B220CODE file *
*            loading. Implemented partial field STA/R/B. *
*                                                        *
* 04/12/16 - Added conditional branches, STx, LDR, LDB,  *
*            LSA, CLx, CLL, SRx, IBB, DBB.               *
*            Revised manual (keyboard) control.          *
*                                                        *
* 04/13/16 - Added non-BCD digit checking for addresses. *
*            Improved macros for B220 code assembly.     *
*            Split source into small 'put' files.        *
*                                                        *
* 04/15/16 - Added SLx and tested all shifts.            *
*                                                        *
* 04/18/16 - Added ADD and SUB and variants.             *
*                                                        *
* 04/19/16 - Added ADL, tested ADD, ADA, SUB, SUA, ADL.  *
*                                                        *
* 04/22/16 - Added simple MUL and a faster, byte-shifting*
*            version (currently FMU).                    *
*                                                        *
* 04/26/16 - Added EXT and RND. Added special cases for  *
*            SRT 10 and  SLT 10.                         *
*                                                        *
* 04/27/16 - Added simple version of DIV.                *
*                                                        *
* 04/29/16 - Added CFA, CFR.                             *
*                                                        *
* 05/02/16 - Added BFA, BFR. Made 'compare' subroutine.  *
*                                                        *
* 05/04/16 - Added RTF, DFL, and DLB.  Split B220EXEC.   *
*                                                        *
* 05/09/16 - Added help redisplay. Paginated EXEC1 & 2.  *
*                                                        *
* 05/12/16 - Moved HLT execution to 'fetch'. Looks good! *
*                                                        *
* 05/15/16 - Fixed bug in 'compare'.  Added simple SPO.  *
*                                                        *
* 05/16/16 - Added Z reset command, revised help.        *
*                                                        *
* 05/18/16 - Added PWR command; first disk command.      *
*                                                        *
* 06/02/16 - Added PRD, PRB commands, removed B220CODE   *
*            pre-load hack.                              *
*                                                        *
* 06/07/16 - Moved FP ops to B220EXEC2. Changed Quit to  *
*            go to full text window and reconnect ProDOS.*
*                                                        *
* 06/19/16 - Fixed STR/STB partial field bug.            *
*                                                        *
* 06/24/16 - Changed PWR to truncate preexisting file.   *
*                                                        *
```

===== Page 3 =====

```
* 07/01/16 - Added FAD, FSU.                         *
*                                                    *
* 07/21/16 - Added FMU.                              *
*                                                    *
* 07/25/16 - Many small JMP --> Bxx space optimizations. *
*            RTF now moves upward! Generalized 'clear'.  *
*                                                    *
* 07/28/16 - Added FDV. Organized shift subroutines.     *
*                                                    *
* 08/22/16 - Modified 'b220asc' table for ) and %.       *
*                                                    *
* 08/27/16 - Fixed LBC bug--hi byte was high by one.     *
*            Fixed SPO: +, form feed, and 'ignore'.      *
*                                                    *
* 09/01/16 - Implemented B220 "tab" in SPO.              *
*                                                    *
* 09/02/16 - Fixed RTF: rB now incremented when NN = 00. *
*                                                    *
* 09/03/16 - Fixed BCH. Was branching on equal.          *
*                                                    *
* 09/05/16 - Fixed IFL, DFL, DLB: if s odd, zeroed s+1.  *
*                                                    *
* 09/09/16 - Added SOR/SOH op and subset of Mag Tape ops.*
*                                                    *
* 09/10/16 - Split PTUNITn into PTRDRn and PTPCHn.       *
*                                                    *
* 09/11/16 - Combined paper tape and mag tape I/O.       *
*                                                    *
* 09/16/16 - Added MRD B-modification.                   *
*                                                    *
* 09/20/16 - Added MPE as NOP.                           *
*                                                    *
* 09/21/16 - Added MLS for SNAP 1E.                      *
*                                                    *
* 09/23/16 - Added IOM (Interrogate Overflow Mode).      *
*                                                    *
* 09/24/16 - Fixed IFL bug: No Ov if hi field posn even. *
*                                                    *
* 11/12/16 - Several small cleanups. ** RELEASED v1.0 ** *
*                                                    *
* 01/16/17 - Moved MEM to top in prep for IOCFG addition.*
*                                                    *
* 01/17/17 - Added I/O configuration editor.             *
*            Restricted PTRDR and PTPCH units to 0 and 1.*
*                                                    *
* 01/25/17 - Integrated I/O Config Editor into B220SIM.  *
*            Fixed MPB bug.                              *
*                                                    *
* 02/01/17 - Added "v1.1" and I/O Config help line.      *
*            ** RELEASED v1.1 **                         *
*                                                    *
* 04/27/17 - Added 'skipincP' to skip P reg increment if *
*            PRB sign 6/7 instruction executed.          *
*                                                    *
* 05/01/17 - Char code matched to CCONV: 04 = ), 10 = (, *
*            27 = $, 32 = ?, 34 = '                      *
*                                                    *
* 06/27/17 - Fixed bug in 'divide', now RTS on overflow. *
*                                                    *
* 08/09/20 - Fixed align & normalization bugs in FAD/FSU.*
*            Fixed post-normalization bug in FDV.        *
*            Kluged KAD as a HLT for rA modification.    *
*            Added "Quit to BASIC" to help lines.        *
*            Cleaned up SUB code.                        *
*                                                    *
* 08/11/20 - Fixed sign logic bugs in CAD/CAA/CSU/CSA.   *
*                                                    *
* 08/12/20 - Fixed rotate bugs in SLA/SLT/SLS.           *
*                                                        *
* 08/13/20 - Preserve rR sign in FDV.                    *
*            Always clear rR in RND.                     *
*            Force rA sign to 0 or 1 in ADL.             *
*            Post-normalize in FAD.                      *
*            Fix FAD result on exponent overflow.        *
*                                                        *
* 08/14/20 - Fix FMU overflow exit state.                *
*                                                        *
* 08/15/20 - Clear rA sign before ovflow check in DIV.   *
*            Clear rA sign if ovflow in FDV.             *
*                                                        *
* 08/16/20 - Force normal zero result in FAD/FSU.        *
*                                                        *
* 08/23/20 - Rewrote SRx to save code!                   *
*                                                        *
* 08/24/20 - Detect EXP Ovflo before mant srT2 in FDV.   *
*            Clear rA EXP on exponent overflow in FDV,   *
*             except when it occurs in ':shrT2'.         *
*            Carry out of mantissa in FAD is not zero.   *
*                                                        *
* 09/01/20 - Changed 'B220msg' to v1.2.                  *
*            Made B220HISTORY a separate PUT file.       *
*            ** RELEASED v1.2 **                         *
*                                                        *
* 09/14/20 - Restarted v2.0 with sim and MEM in Aux mem, *
*             leaving panel, I/O, and display in Main.   *
*            Fixed FMU pending ovflow in FMU & MUL to    *
*             be compatible with SOR mode.               *
*                                                        *
* 09/15/20 - Rewrote ]prd and PWR to use buffered I/O.   *
*                                                        *
* 09/22/20 - Rewrote B220IO to support paper tape.       *
*                                                        *
* 09/26/20 - Paper tape buffered I/O works.              *
*                                                        *
* 10/11/20 - Replaced 'bcdcor' with 'bcd2bin' table.     *
*                                                        *
* 11/23/20 - Integrated B220IO and B220MT for trial.     *
*                                                        *
* 12/04/20 - Version 2.0 now runs REGRESS.OBJ.           *
*            Debugging Mag Tape operations.              *
*                                                        *
* 01/12/21 - Fixed numerous bugs--still testing.         *
*                                                        *
**********************************************************
```

===== Page 5 =====

```
         use   B220DEFS
* 6502 equates

BCSop    equ   $B0        ; BCS opcode
BNEop    equ   $D0        ; BNE opcode
CLCop    equ   $18        ; CLC opcode
SECop    equ   $38        ; SEC opcode
NOPop    equ   $EA        ; NOP opcode
ADCZop   equ   $65        ; ADC zp opcode
BITZop   equ   $24        ; BIT zp opcode
CMPIop   equ   $C9        ; CMP # opcode
SBCZop   equ   $E5        ; SBC zp opcode
ADCYop   equ   $79        ; ADC aaaa,y opcode
SBCYop   equ   $F9        ; SBC aaaa,y opcode

* Apple equates

WNDTOP   equ   $22        ; Top line of text window
CH       equ   $24        ; COUT horizontal cursor
BASL     equ   $28        ; Screen base address
IN       equ   $200       ; Keyboard input buffer
KBD      equ   $C000      ; Keyboard port
READMAIN equ   $C002      ; Store to read Main
READAUX  equ   $C003      ; Store to read Aux
WRITMAIN equ   $C004      ; Store to write Main
WRITAUX  equ   $C005      ; Store to write Aux
ALTCHAR  equ   $C00F      ; Store to enable alt charset
KBSTROBE equ   $C010      ; Keyboard strobe reset
SPKR     equ   $C030      ; Toggle speaker

* Apple entry points

DOSCON   equ   $3D0       ; ProDOS reconnect vector
DOSCMD   equ   $BE03      ; BASIC.SYSTEM PDOS command
PRINTERR equ   $BE0C      ; Print ProDOS error msg
BSSTATE  equ   $BE42      ; BASIC.SYSTEM state var
PRBL2    equ   $F94A      ; Print (X) blanks
TABV     equ   $FB5B      ; Vertical tab to (A)
BASCALC  equ   $FBC1      ; Set BASL to line (A)
BEEP     equ   $FBDD      ; Beep
HOME     equ   $FC58      ; Clear screen
CROUT    equ   $FD8E      ; Output a CR
COUT     equ   $FDED      ; Output char in A

* Simulation parameters

memb     equ   5000*6     ; 5000 6-byte B220 words
MEM      equ   $C000-memb ; Simulated B220 memory in Aux
ndb      equ   6          ; Number of Device Blocks
dispcnt  equ   100        ; Update panel every 100 instrs

* Buffered I/O flag byte definitions

EOF      equ   $EF        ; End-Of-File flag byte
EMPTY    equ   $EE        ; Empty buffer flag byte
EOB      equ   $EB        ; End-Of-Buffer flag byte
PREF     equ   $B0        ; Block prefix sign flag
```

```
                ************************************************************
                *                                                          *
                *                 Page zero variables                      *
                *                                                          *
                ************************************************************


                         dum   $90        ; Start of Page Zero variables

                * B220 memory fields

                S        equ   0          ; Sign digit
                sL       equ   1          ; rC sL specifier
                VV       equ   2          ; rC Variant
                OP       equ   3          ; rC Op code
                ADDR     equ   4          ; rC BCD address
                EXP      equ   1          ; FP exponent
                MANT     equ   2          ; FP mantissa

                * Simulated B220 State Variables

                B220strt equ   *          ; Start of simulated B220 state
0090: 00 00 00  rBx      ds    4          ; 4 const zero byte prefix to rB
0094: 00 00     rB       dw    0          ; BCD B register
0096: 00 00     rP       dw    0          ; BCD P register
0098: 00 00 00  rC       ds    6          ; BCD Control (instruction) reg
009E: 00 00 00  rA       ds    6          ; BCD A register
00A4: 00 00 00  rR       ds    6          ; BCD R register
00AA: 00 00 00  rD       ds    6          ; BCD D register
00B0: 00 00 00  rD10     ds    6          ; BCD D10 reg (rD * 10)
00B6: 00 00 00  CSW      ds    10         ; Control switches (0=off)
00C0: 00        RUN      db    0          ; RUN mode/indicator (0=off)
00C1: 00        ERR      db    0          ; ERR indicator (0=off)
00C2: 00        COMP     db    0          ; Compare lo,eql,hi (<0,0,>0)
00C3: 00        Ov       db    0          ; Overflow indicator (0=off)
00C4: 00        Rp       db    0          ; Repeat indicator (0=off)
00C5: 00        newp     db    0          ; "P changed manually" indicator
00C6: 00        skipincP db    0          ; Skip incP if PRB sign 6/7.
                B220end  equ   *          ; End of B220 simulated state

                * Simulator page zero variables

00C7: FF        OvHlt    db    $FF        ; OVerflow Halt toggle (0=off)
00C8: 00 00     instptr  dw    0          ; Pointer corresponding to rP
00CA: 00 00     memptr   dw    0          ; Pointer to instruction data
00CC: 00 00     ptr      dw    0          ; Utility pointer
00CE: 00 00     inptr    dw    0          ; 'keyin' register label ptr
00D0: 00        t1       db    0          ; Temp byte
00D1: 00        NN       db    0          ; 2-digit BCD count
00D2: 00        dbx      db    0          ; Device Block index
00D3: 64        dispctr  db    dispcnt    ; Display refresh counter
00D4: 00 00     linev    dw    0          ; Line base for decimal value
00D6: 00 00     line1    dw    0          ; Line base for 1-bits
00D8: 00 00     line2    dw    0          ; Line base for 2-bits
00DA: 00 00     line4    dw    0          ; Line base for 4-bits
00DC: 00 00     line8    dw    0          ; Line base for 8-bits
                         dend
```

```
**********************************************************
*                                                        *
*                  Macro Definitions                     *
*                                                        *
**********************************************************

auxjmp   mac               ; <addr>
         sta   READAUX
         sta   WRITAUX
         jmp   ]1
         eom

auxjsr   mac               ; <addr>
         sta   READAUX
         sta   WRITAUX
         jsr   ]1
         sta   READMAIN
         sta   WRITMAIN
         rts
         eom

mainjmp  mac               ; <addr>
         sta   READMAIN
         sta   WRITMAIN
         jmp   ]1
         eom

mainjsr  mac               ; <addr>
         sta   READMAIN
         sta   WRITMAIN
         jsr   ]1
         jmp   AUXrts
         eom

seti     mac               ; Set indicator
         lda   #$FF
         sta   ]1          ; Set non-zero.
         eom

resi     mac               ; Reset indicator
         lda   #0
         sta   ]1          ; Zero indicator.
         eom

align    mac
         ds    *-1/]1*]1+]1-*
         eom
```

```
                59             dsk   /ap/merlin/work/b220/b220sim
                60
                61             org   $800       ; Start of Main code
                62             put   B220COMMON
               >1    **********************************************************
               >2    *                                                        *
               >3    *         B220SIM Common Code (Main and Auxmem)          *
               >4    *                                                        *
               >5    **********************************************************
               >6
               >7    common   equ   *          ; Start of code common to Aux & Main
               >8
               >9    * Entry point and restart vector
               >10
0800: 4C D7 08 >11   B220SIM  jmp   init       ; Initialize simulation
0803: 4C 47 09 >12   RESTART  jmp   restart    ; Restart warm.
               >13
               >14   * Vectors for Main to reference Auxmem routines
               >15
               >16   X_fetch  auxjmp fetch
0806: 8D 03 C0 >16            sta   READAUX
0809: 8D 05 C0 >16            sta   WRITAUX
080C: 4C 01 09 >16            jmp   fetch
               >16            eom
               >17   X_newP   auxjmp newP
080F: 8D 03 C0 >17            sta   READAUX
0812: 8D 05 C0 >17            sta   WRITAUX
0815: 4C E3 08 >17            jmp   newP
               >17            eom
               >18   X_cont   auxjmp ]contin
0818: 8D 03 C0 >18            sta   READAUX
081B: 8D 05 C0 >18            sta   WRITAUX
081E: 4C 50 09 >18            jmp   ]contin
               >18            eom
               >19   X_IOerr  auxjmp IOerr
0821: 8D 03 C0 >19            sta   READAUX
0824: 8D 05 C0 >19            sta   WRITAUX
0827: 4C D3 09 >19            jmp   IOerr
               >19            eom
               >20
               >21   X_incP   auxjsr incP
082A: 8D 03 C0 >21            sta   READAUX
082D: 8D 05 C0 >21            sta   WRITAUX
0830: 20 A3 09 >21            jsr   incP
0833: 8D 02 C0 >21            sta   READMAIN
0836: 8D 04 C0 >21            sta   WRITMAIN
0839: 60       >21            rts
               >21            eom
               >23   * Vectors for Aux to reference Main routines
               >24
               >25   M_keyin  mainjmp keyin
083A: 8D 02 C0 >25            sta   READMAIN
083D: 8D 04 C0 >25            sta   WRITMAIN
0840: 4C 55 09 >25            jmp   keyin
               >25            eom
               >26   M_stop   mainjmp ]stop
0843: 8D 02 C0 >26            sta   READMAIN
0846: 8D 04 C0 >26            sta   WRITMAIN
0849: 4C 5C 09 >26            jmp   ]stop
               >26            eom
               >27
               >28   M_disp   mainjsr display
084C: 8D 02 C0 >28            sta   READMAIN
084F: 8D 04 C0 >28            sta   WRITMAIN
0852: 20 33 0F >28            jsr   display
0855: 4C C4 08 >28            jmp   AUXrts
               >28            eom
               >29   M_iosel  mainjsr iosel
0858: 8D 02 C0 >29            sta   READMAIN
085B: 8D 04 C0 >29            sta   WRITMAIN
085E: 20 C8 11 >29            jsr   iosel
0861: 4C C4 08 >29            jmp   AUXrts
               >29            eom
               >30   M_iodsel mainjsr iodsel
0864: 8D 02 C0 >30            sta   READMAIN
0867: 8D 04 C0 >30            sta   WRITMAIN
086A: 20 EC 11 >30            jsr   iodsel
086D: 4C C4 08 >30            jmp   AUXrts
               >30            eom
               >31   M_getwrd mainjsr getwrd
0870: 8D 02 C0 >31            sta   READMAIN
0873: 8D 04 C0 >31            sta   WRITMAIN
0876: 20 F9 11 >31            jsr   getwrd
0879: 4C C4 08 >31            jmp   AUXrts
               >31            eom
               >32   M_putwrd mainjsr putwrd
087C: 8D 02 C0 >32            sta   READMAIN
087F: 8D 04 C0 >32            sta   WRITMAIN
0882: 20 32 12 >32            jsr   putwrd
0885: 4C C4 08 >32            jmp   AUXrts
               >32            eom
               >33   M_setlan mainjsr setlan
0888: 8D 02 C0 >33            sta   READMAIN
088B: 8D 04 C0 >33            sta   WRITMAIN
088E: 20 4B 13 >33            jsr   setlan
0891: 4C C4 08 >33            jmp   AUXrts
               >33            eom
               >34   M_resetd mainjsr resetdb
0894: 8D 02 C0 >34            sta   READMAIN
0897: 8D 04 C0 >34            sta   WRITMAIN
089A: 20 84 13 >34            jsr   resetdb
089D: 4C C4 08 >34            jmp   AUXrts
               >34            eom
               >35   M_nxtblk mainjsr nxtblk
08A0: 8D 02 C0 >35            sta   READMAIN
08A3: 8D 04 C0 >35            sta   WRITMAIN
08A6: 20 7E 12 >35            jsr   nxtblk
08A9: 4C C4 08 >35            jmp   AUXrts
               >35            eom
               >36   M_prvblk mainjsr prvblk
08AC: 8D 02 C0 >36            sta   READMAIN
08AF: 8D 04 C0 >36            sta   WRITMAIN
08B2: 20 D2 12 >36            jsr   prvblk
08B5: 4C C4 08 >36            jmp   AUXrts
               >36            eom
               >37   M_COUT   mainjsr COUT
08B8: 8D 02 C0 >37            sta   READMAIN
08BB: 8D 04 C0 >37            sta   WRITMAIN
08BE: 20 ED FD >37            jsr   COUT
08C1: 4C C4 08 >37            jmp   AUXrts
               >37            eom
               >38
08C4: 8D 03 C0 >39   AUXrts   sta   READAUX
08C7: 8D 05 C0 >40            sta   WRITAUX
08CA: 60       >41            rts
```

```
               * Subroutines duplicated in both Aux and Main

08CB: 18       incmem   clc              ; Advance memptr
08CC: A5 CA             lda   memptr     ;  to next word.
08CE: 69 06             adc   #6
08D0: 85 CA             sta   memptr
08D2: 90 02             bcc   :nocarry
08D4: E6 CB             inc   memptr+1   ; Propagate carry.
08D6: 60       :nocarry rts

               endcomm  equ   *          ; End of code common to Aux & Main
                        err   endcomm-common/256 ; Must be < 1 page.
```

```
                              ===== Page 12 =====

                 63            put   B220INIT
                >1     ***********************************************************
                >2     *                                                         *
                >3     *                    Initialize B220                      *
                >4     *                                                         *
                >5     ***********************************************************
                >6
08D7: AD 54 09 >7      init     lda   initstk    ; Been here before?
08DA: D0 3D    >8               bne   :notinit   ; -Yes, skip init copys.
08DC: BA       >9               tsx              ; -No, save initial stk ptr.
08DD: 8E 54 09 >10              stx   initstk
08E0: A9 00    >11              lda   #<common   ; Copy common code from Main-->Aux
08E2: 85 CA    >12              sta   memptr
08E4: A9 08    >13              lda   #>common
08E6: 85 CB    >14              sta   memptr+1
08E8: A0 D6    >15              ldy   #endcomm-common-1
08EA: 8D 05 C0 >16              sta   WRITAUX    ; Stores go to AUX memory.
08ED: B1 CA    >17     :comlp   lda   (memptr),y
08EF: 91 CA    >18              sta   (memptr),y
08F1: 88       >19              dey
08F2: C0 FF    >20              cpy   #$FF       ; Underflow?
08F4: D0 F7    >21              bne   :comlp     ; -No, keep copying...
08F6: A9 3D    >22              lda   #<AUXcode  ; Copy B220SIM to Aux mem
08F8: 85 CC    >23              sta   ptr
08FA: A9 15    >24              lda   #>AUXcode
08FC: 85 CD    >25              sta   ptr+1
08FE: A9 D7    >26              lda   #<endcomm
0900: 85 CA    >27              sta   memptr
0902: A9 08    >28              lda   #>endcomm
0904: 85 CB    >29              sta   memptr+1
0906: A0 00    >30              ldy   #0         ; Move a page
0908: B1 CC    >31     :auxlp   lda   (ptr),y
090A: 91 CA    >32              sta   (memptr),y
090C: C8       >33              iny
090D: D0 F9    >34              bne   :auxlp
090F: E6 CB    >35              inc   memptr+1
0911: E6 CD    >36              inc   ptr+1
0913: A5 CD    >37              lda   ptr+1
0915: C9 28    >38              cmp   #>AUXend+$100 ; Past last page?
0917: 90 EF    >39              bcc   :auxlp     ; -No, keep moving.
0919: 8D 05 C0 >40     :notinit sta   WRITAUX    ; Stores go to AUX memory.
091C: A9 D0    >41              lda   #<MEM      ; Initialize B220 memory to 0
091E: 85 CA    >42              sta   memptr
0920: A9 4A    >43              lda   #>MEM
0922: 85 CB    >44              sta   memptr+1
0924: A0 00    >45              ldy   #0
0926: 98       >46     :loop    tya
0927: 91 CA    >47     :pagloop sta   (memptr),y
0929: C8       >48              iny
092A: D0 FB    >49              bne   :pagloop
092C: E6 CB    >50              inc   memptr+1
092E: A5 CB    >51              lda   memptr+1
0930: C9 96    >52              cmp   #>$9600
0932: 90 F2    >53              bcc   :loop
0934: 8D 04 C0 >54              sta   WRITMAIN   ; Back to Main mem
0937: A2 36    >55     reset    ldx   #B220end-B220strt-1 ; Clear B220 state
0939: A9 00    >56              lda   #0
093B: 95 90    >57     :regloop sta   B220strt,x
093D: CA       >58              dex
093E: 10 FB    >59              bpl   :regloop
0940: 20 75 13 >60              jsr   resetdbs   ; Rewind all tapes.
               >61              seti  OvHlt      ; Set Ovflow Halt mode.
0943: A9 FF    >61              lda   #$FF
0945: 85 C7    >61              sta   OvHlt      ; Set non-zero.
               >61              eom
0947: AE 54 09 >62     restart  ldx   initstk    ; Restore initial stack ptr.
```

```
094A: 9A                       txs
094B: 20 04 0D                 jsr   disppanl   ; Init screen for B220
094E: 20 33 0F                 jsr   display    ;  panel & display state.
0951: 4C 0F 08                 jmp   X_newP     ; Start simulation.

0954: 00             initstk   db    0          ; Stack pointer at entry.
```

```
                64            put   B220KEYB
               >1    **********************************************************
               >2    *                                                        *
               >3    *                Keyboard Input Routines                 *
               >4    *                                                        *
               >5    **********************************************************
               >6
0955: 8D 10 C0 >7    keyin    sta   KBSTROBE   ; Clear strobe.
0958: C9 A0    >8             cmp   #"         " ; Space bar?
095A: D0 49    >9             bne   :bleep     ; -No, beep & continue.
               >10   ]stop    resi  RUN        ; -Yes, reset RUN mode
095C: A9 00    >10            lda   #0
095E: 85 C0    >10            sta   RUN        ; Zero indicator.
               >10            eom
0960: AD 67 05 >11            lda   ERRlab     ; Did I/O error
0963: C9 C9    >12            cmp   #"I"       ;  get us here?
0965: F0 03    >13            beq   :edit      ; -Yes, don't flush.
0967: 20 D0 13 >14            jsr   flushall   ; -No, flush all buffers.
096A: 20 33 0F >15   :edit    jsr   display    ; Update B220 panel
               >16            resi  ERR        ; Reset ERR indicator
096D: A9 00    >16            lda   #0
096F: 85 C1    >16            sta   ERR        ; Zero indicator.
               >16            eom
0971: AD 00 C0 >17   :waitkey lda   KBD        ; Get a key.
0974: 10 FB    >18            bpl   :waitkey
0976: 8D 10 C0 >19            sta   KBSTROBE   ; Clear strobe
0979: C9 A0    >20            cmp   #"         " ; Space bar?
097B: F0 0E    >21            beq   :step      ; -Yes, step.
097D: C9 BF    >22            cmp   #"?"       ; Show help?
097F: F0 5F    >23            beq   :disphlp   ; -Yes, do it.
0981: 29 DF    >24            and   #$DF       ; Force upper case.
0983: C9 C7    >25            cmp   #"G"       ; G = Go?
0985: D0 24    >26            bne   :nx1       ; -No, analyze keypress.
               >27            seti  RUN        ; -Yes, set RUN mode
0987: A9 FF    >27            lda   #$FF
0989: 85 C0    >27            sta   RUN        ; Set non-zero.
               >27            eom
098B: A9 F2    >28   :step    lda   #"r"       ; Reset ERRlab on screen
098D: 8D 67 05 >29            sta   ERRlab
0990: A5 C5    >30            lda   newp       ; rP changed manually?
0992: D0 0A    >31            bne   :new       ; -Yes, re-fetch.
0994: A5 9B    >32            lda   rC+OP      ; -No, is OP a HLT?
0996: D0 10    >33            bne   :xeq       ; -No, execute current OP
0998: 20 2A 08 >34            jsr   X_incP     ; -Yes, skip HLT
099B: 4C 06 08 >35            jmp   X_fetch    ;   and fetch next.
               >36
               >37   :new     resi  newp       ; Reset new P indicator
099E: A9 00    >37            lda   #0
09A0: 85 C5    >37            sta   newp       ; Zero indicator.
               >37            eom
09A2: 4C 0F 08 >38            jmp   X_newP     ;  and re-fetch.
               >39
09A5: 20 DD FB >40   :bleep   jsr   BEEP       ; Beep
09A8: 4C 18 08 >41   :xeq     jmp   X_cont     ; Execute current OP.
               >42
09AB: C9 D1    >43   :nx1     cmp   #"Q"       ; Quit?
09AD: D0 0B    >44            bne   :nx2       ; -No, continue.
09AF: D8       >45            cld              ; -Yes, clear decimal
09B0: 18       >46            clc              ;   and Carry.
09B1: A9 00    >47            lda   #0         ; Set full-screen
09B3: 85 22    >48            sta   WNDTOP     ;  text window,
09B5: 68       >49            pla              ;  pop return
09B6: 68       >50            pla              ;   address, and
09B7: 4C D0 03 >51            jmp   DOSCON     ;    reconnect ProDOS.
               >52
09BA: C9 D3    >53   :nx2     cmp   #"S"       ; Toggle switch?
09BC: F0 28    >54            beq   :flipsw    ; -Yes.
```

===== Page 15 =====

```
09BE: C9 C1    >55           cmp   #"A"       ; A register?
09C0: F0 64    >56           beq   :inputA    ; -Yes, get input.
09C2: C9 D2    >57           cmp   #"R"       ; R register?
09C4: F0 64    >58           beq   :inputR    ; -Yes, get input.
09C6: C9 C2    >59           cmp   #"B"       ; B register?
09C8: F0 64    >60           beq   :inputB    ; -Yes, get input.
09CA: C9 D0    >61           cmp   #"P"       ; P register?
09CC: F0 68    >62           beq   :inputP    ; -Yes, get input.
09CE: C9 C3    >63           cmp   #"C"       ; C register?
09D0: F0 60    >64           beq   :inputC    ; -Yes, get input.
09D2: C9 DA    >65           cmp   #"Z"       ; Reset?
09D4: F0 39    >66           beq   :reset     ; -Yes, clear state.
09D6: C9 C9    >67           cmp   #"I"       ; I/O configuration?
09D8: F0 3F    >68           beq   :edio      ; -Yes, edit I/O config.
09DA: 20 DD FB >69   :beep    jsr   BEEP       ; Unrecognized key, beep
09DD: 4C 71 09 >70           jmp   :waitkey   ;  and get another key.
               >71
09E0: 20 22 0F >72   :disphlp jsr   disphelp   ; Display help lines
09E3: 4C 71 09 >73           jmp   :waitkey   ;  and get another key.
               >74
09E6: A9 13    >75   :flipsw  lda   #$13       ; Set "Sw" label to inverse.
09E8: 8D 53 05 >76           sta   SWlab
09EB: A9 77    >77           lda   #$77
09ED: 8D 54 05 >78           sta   SWlab+1
09F0: 20 B0 0A >79           jsr   getdig     ; Get digit or CR
09F3: B0 0D    >80           bcs   :swdone    ; Done if CR.
09F5: AA       >81           tax              ; -No, handle digit.
09F6: B5 B6    >82           lda   CSW,x      ; Pick up switch,
09F8: F0 04    >83           beq   :seti      ; -If reset, set it.
09FA: A9 00    >84           lda   #0         ; -If set, reset it.
09FC: F0 02    >85           beq   :store     ; (always)
               >86
09FE: A9 FF    >87   :seti    lda   #$FF
0A00: 95 B6    >88   :store   sta   CSW,x      ;   put it back.
0A02: A9 D3    >89   :swdone  lda   #"S"       ; Set "Sw" label to normal.
0A04: 8D 53 05 >90           sta   SWlab
0A07: A9 F7    >91           lda   #"w"
0A09: 8D 54 05 >92           sta   SWlab+1
0A0C: 4C 6A 09 >93   :ed      jmp   :edit
               >94
0A0F: 20 37 09 >95   :reset   jsr   reset      ; Reset B220 state
               >96           seti  newp       ; Force refetch.
0A12: A9 FF    >96           lda   #$FF
0A14: 85 C5    >96           sta   newp       ; Set non-zero.
               >96           eom
0A16: 4C 0C 0A >97           jmp   :ed
               >98
0A19: 4C 6D 0B >99   :edio    jmp   ediocfg    ; Relay jump
               >100
0A1C: A0 00    >101  :indone  ldy   #0         ; Flip reg label to normal.
0A1E: B1 CE    >102          lda   (inptr),y
0A20: 09 80    >103          ora   #$80
0A22: 91 CE    >104          sta   (inptr),y
0A24: D0 E6    >105          bne   :ed        ; (always)
               >106
0A26: A2 00    >107  :inputA  ldx   #Ain-intabl
0A28: B0 12    >108          bcs   :inreg     ; (always)
               >109
0A2A: A2 10    >110  :inputR  ldx   #Rin-intabl
0A2C: B0 0E    >111          bcs   :inreg     ; (always)
               >112
0A2E: A2 04    >113  :inputB  ldx   #Bin-intabl
0A30: B0 0A    >114          bcs   :inreg     ; (always)
               >115
0A32: A2 08    >116  :inputC  ldx   #Cin-intabl
0A34: B0 06    >117          bcs   :inreg     ; (always)
               >118
```

```
0A36: A2 0C    >119  :inputP  ldx   #Pin-intabl
               >120           seti  newp       ; Signal manual rP change.
0A38: A9 FF    >120           lda   #$FF
0A3A: 85 C5    >120           sta   newp       ; Set non-zero.
               >120           eom
               >121  *                         ;  and fall into :inreg.
               >122
               >123  * Input register value from keyboard
               >124  *   On entry: X = intabl index
               >125  *   On exit:  Y = Hi (left) byte of register
               >126  *             X = # of bytes in register - 1
               >127
0A3C: BD 75 0A >128  :inreg   lda   intabl,x   ; Set inptr to reg label
0A3F: 85 CE    >129           sta   inptr
0A41: BD 76 0A >130           lda   intabl+1,x
0A44: 85 CF    >131           sta   inptr+1
0A46: BC 77 0A >132           ldy   intabl+2,x ; Y = hi byte of reg
0A49: 8C 69 0A >133           sty   :ordig+1   ; Save register address
0A4C: 8C 6B 0A >134           sty   :stdig+1
0A4F: BD 78 0A >135           lda   intabl+3,x
0A52: AA       >136           tax              ; X = reg length - 1
0A53: A0 00    >137           ldy   #0
0A55: B1 CE    >138           lda   (inptr),y  ; Flip reg label to inverse.
0A57: 29 7F    >139           and   #$7F
0A59: 91 CE    >140           sta   (inptr),y
0A5B: 20 B0 0A >141  :getdig  jsr   getdig     ; Get digit or CR
0A5E: B0 BC    >142           bcs   :indone    ; CR ==> done.
0A60: 48       >143           pha              ; Save digit
0A61: AC 69 0A >144           ldy   :ordig+1   ; Restore Y
0A64: 20 89 0A >145           jsr   shleft1    ; Shift register left 1 digit
0A67: 68       >146           pla              ; Recover the digit
0A68: 15 00    >147  :ordig   ora   0*0,x      ; OR in the low digit
0A6A: 95 00    >148  :stdig   sta   0*0,x      ;  and store it back.
0A6C: 8A       >149           txa              ; Save X
0A6D: 48       >150           pha
0A6E: 20 33 0F >151           jsr   display    ; Update display
0A71: 68       >152           pla              ; Restore X
0A72: AA       >153           tax
0A73: D0 E6    >154           bne   :getdig    ; (always)
               >155
               >156  intabl   equ   *          ; Table of reg input params
0A75: 83 05    >157  Ain      dw    Alab       ; Address of "A" label
0A77: 9E 05    >158           db    rA,6-1     ; Addr of hi digit, length-1
0A79: AB 05    >159  Bin      dw    Blab
0A7B: 94 01    >160           db    rB,2-1
0A7D: BB 05    >161  Cin      dw    Clab
0A7F: 98 05    >162           db    rC,6-1
0A81: B3 05    >163  Pin      dw    Plab
0A83: 96 01    >164           db    rP,2-1
0A85: 95 05    >165  Rin      dw    Rlab
0A87: A4 05    >166           db    rR,6-1
```

```
                ****************************************************
                *                                                        *
                *         Shift Register left 1 digit (4 bits)           *
                *                                                        *
                * On entry: Y = addr of Hi (left) byte of register       *
                *           X = register byte length - 1                 *
                *                                                        *
                * On exit:  X and Y are unchanged.  If rA, rR, or rC,    *
                *           the high digit of the sign byte is cleared.  *
                *                                                        *
                **********************************************************

0A89: 8C 91 0A  shleft1  sty   :shift+1   ; Save register address
0A8C: 8A                 txa              ;  and byte length - 1.
0A8D: A0 04              ldy   #4         ; Digit = 4 bits.
0A8F: 18        :nxshift clc              ; Shift in zeroes.
0A90: 36 00     :shift   rol   0*0,x      ; Shift 1 bit
0A92: CA                 dex              ;  for all bytes.
0A93: 10 FB              bpl   :shift
0A95: AA                 tax              ; Restore X
0A96: 88                 dey
0A97: D0 F6              bne   :nxshift   ; Shift 4 times.
0A99: AC 91 0A           ldy   :shift+1   ; Restore Y = reg address.
0A9C: C0 96              cpy   #rP        ; rP has no sign byte,
0A9E: F0 0C              beq   :rts       ;  so skip it.
0AA0: C0 94              cpy   #rB        ; rB has no sign byte,
0AA2: F0 08              beq   :rts       ;  so skip it.
0AA4: B9 00 00           lda   0,y        ; Clear high digit
0AA7: 29 0F              and   #$0F       ;  of sign byte.
0AA9: 99 00 00           sta   0,y
0AAC: 60        :rts     rts

                **********************************************************
                *                                                        *
                *                   Get Digit or CR                      *
                *                                                        *
                * On exit: If C = 0, A = digit value                     *
                *          If C = 1, CR received                         *
                *          X and Y unchanged.                            *
                *                                                        *
                **********************************************************

0AAD: 20 DD FB  beepget  jsr   BEEP       ; Signal error key
0AB0: AD 00 C0  getdig   lda   KBD        ; Get digit or <Enter>
0AB3: 10 FB              bpl   getdig
0AB5: 8D 10 C0           sta   KBSTROBE   ; Clear strobe
0AB8: C9 8D              cmp   #$8D       ; <Enter>?
0ABA: F0 0A              beq   :done      ; Yes, exit.
0ABC: C9 B0              cmp   #"0"       ; -No, less than "0"?
0ABE: 90 ED              bcc   beepget    ; -Yes, get another.
0AC0: C9 BA              cmp   #"9"+1     ; -No, greater than "9"?
0AC2: B0 E9              bcs   beepget    ; -Yes, get another.
0AC4: 29 0F              and   #$0F       ; -No, isolate digit
0AC6: 60        :done    rts              ; C ==> digit, /C ==> CR.
```

```
                              ===== Page 18 =====

               **********************************************************
               *                                                        *
               *            Edit B220SIM I/O Configuration              *
               *                                                        *
               **********************************************************

               cursor   equ   $57        ; Mousetext checkerboard
               uparrow  equ   $8B        ; Up arrow
               dnarrow  equ   $8A        ; Down arrow
               ltarrow  equ   $88        ; Left arrow
               escape   equ   $9B        ; ESCAPE key
               delete   equ   $FF        ; DELETE key
               iocfgtt  equ   11         ; HTAB for screen title
               rtmargin equ   4          ; Right margin
               fnamecol equ   rtmargin+8 ; File name column

               fnx      equ   linev      ; File name index (0..7)
               selected equ   linev+1    ; Selected index (0..7)
               selsave  equ   line1      ; Temp Y storage
               savex    equ   line1+1    ; Temp X storage
               selch    equ   line2      ; Selected fname cursor
               line     equ   line2+1    ; Line number (0..23)
               changed  equ   line4      ; File name changed flg
               selBASL  equ   line8      ; Selected line base (DA.DB)

               iocfgstr equ   *          ; I/O Config Screen string
0AC7: C9 AF CF          asc   "I/O Configuration",0D
0AD9: 0D                db    $0D
0ADA: A0 D5 EE          asc   " Unit    File pathname",0D
0AF1: AD AD AD          asc   "------  ------------------------",0D
0B12: D0 D4 D2          asc   "PTRDR0",01
0B19: D0 D4 D2          asc   "PTRDR1",01
0B20: D0 D4 D0          asc   "PTPCH0",01
0B27: D0 D4 D0          asc   "PTPCH1",01
0B2E: 0D                db    $0D
0B2F: CD D4 D5          asc   "MTU0L0",01
0B36: CD D4 D5          asc   "MTU0L1",01
0B3D: CD D4 D5          asc   "MTU1L0",01
0B44: CD D4 D5          asc   "MTU1L1",01
0B4B: 0D 0D 0D          db    $0D,$0D,$0D,$0D,$0D
0B50: A0 A0 A0          asc   "    ESC to return to B220SIM"
0B6C: 00                db    00         ; End of screen

0B6D: A2 00    ediocfg  ldx   #0         ; Edit I/O Configuration
0B6F: 86 22             stx   WNDTOP     ; Set full screen.
0B71: 86 D5             stx   selected   ; Select first file name.
0B73: 20 58 FC          jsr   HOME       ; Clear screen
0B76: A2 00    disiocfg ldx   #0         ; iocfgstr index = 0
0B78: 86 D4             stx   fnx        ; fname index = 0
0B7A: 86 D9             stx   line       ; Line = 0
0B7C: 8A                txa
0B7D: 20 C1 FB          jsr   BASCALC    ; Set BASL for line 0
0B80: A0 0B             ldy   #iocfgtt   ; HTAB to title
0B82: BD C7 0A :nxch    lda   iocfgstr,x ; Next disp string char
0B85: 10 06             bpl   :command   ; -Command char if +
0B87: 91 28             sta   (BASL),y   ; -Display if not cmd.
0B89: C8                iny              ; Advance CH
0B8A: E8       :advance inx              ; Advance str index
0B8B: D0 F5             bne   :nxch      ; (always)

0B8D: F0 48    :command beq   :editfn    ; Screen complete, edit.
0B8F: C9 0D             cmp   #$0D       ; CR?
0B91: D0 0B             bne   :fname     ; -No, insert file name.
0B93: E6 D9    :nxtline inc   line       ; -Yes, next line.
0B95: A5 D9             lda   line       ; Compute new line's
0B97: 20 C1 FB          jsr   BASCALC    ;  base addr (BASL)
0B9A: A0 04             ldy   #rtmargin  ; Set right margin.
```

```
0B9C: 10 EC    >290           bpl   :advance   ; (always)
               >291
0B9E: 86 D7    >292  :fname   stx   savex      ; Insert file name.
0BA0: A9 BA    >293           lda   #":"       ; Insert punctuation.
0BA2: 91 28    >294           sta   (BASL),y
0BA4: A4 D4    >295           ldy   fnx
0BA6: C4 D5    >296           cpy   selected   ; This fname selected?
0BA8: F0 01    >297           beq   :selectd   ; -Yes, C = selected.
0BAA: 18       >298           clc              ; -No, /C = not selected.
0BAB: BE C6 10 >299  :selectd ldx   fnxfn,y    ; Index into fnames
0BAE: A0 0C    >300           ldy   #fnamecol  ; Y = 1st char of filename.
0BB0: BD 00 11 >301  :nxtchar lda   fnames,x   ; Next file name char.
0BB3: F0 0C    >302           beq   :fndone    ; End of file name.
0BB5: 90 04    >303           bcc   :store     ; /C ==> keep normal.
0BB7: 20 8D 0C >304           jsr   inverse    ; C ==> make inverse
0BBA: 38       >305           sec              ;  and stay selected.
0BBB: 91 28    >306  :store   sta   (BASL),y   ; Display character
0BBD: E8       >307           inx              ; Advance fnames index
0BBE: C8       >308           iny              ; Advance CH
0BBF: D0 EF    >309           bne   :nxtchar   ; (always)
               >310
0BC1: E6 D4    >311  :fndone  inc   fnx        ; Advance fnames index
0BC3: A6 D7    >312           ldx   savex      ; Restore string index
0BC5: 90 CC    >313           bcc   :nxtline   ; Not selected ==> done.
0BC7: A9 57    >314           lda   #cursor    ; Selected ==> add cursor.
0BC9: 91 28    >315           sta   (BASL),y
0BCB: 84 D8    >316           sty   selch      ; Save cursor column.
0BCD: A5 28    >317           lda   BASL       ; Save selected line base
0BCF: 85 DC    >318           sta   selBASL
0BD1: A5 29    >319           lda   BASL+1
0BD3: 85 DD    >320           sta   selBASL+1
0BD5: D0 BC    >321           bne   :nxtline   ; (always)
               >322
0BD7: A4 D8    >323  :editfn  ldy   selch      ; Cursor col of selected.
0BD9: A9 00    >324           lda   #0         ; Mark unchanged.
0BDB: 85 DA    >325           sta   changed
0BDD: AD 00 C0 >326  :kbdloop lda   KBD        ; Read key and
0BE0: 10 FB    >327           bpl   :kbdloop   ;  wait for keypress.
0BE2: 8D 10 C0 >328           sta   KBSTROBE   ; Clear keyboard strobe.
0BE5: A6 D5    >329           ldx   selected   ; Save index of currently
0BE7: 86 D6    >330           stx   selsave    ;  selected file name.
0BE9: C9 8B    >331           cmp   #uparrow
0BEB: D0 52    >332           bne   :notup
0BED: C6 D5    >333           dec   selected   ; Move cursor up
0BEF: A5 D5    >334           lda   selected   ;  and wrap around.
0BF1: 29 07    >335           and   #7
0BF3: 85 D5    >336           sta   selected
0BF5: A9 A0    >337  :edited  lda   #"         " ; Blank out cursor
0BF7: A4 D8    >338           ldy   selch
0BF9: 91 DC    >339           sta   (selBASL),y
0BFB: A5 DA    >340           lda   changed    ; Fname changed?
0BFD: F0 29    >341           beq   :chkexit   ; -No, exit or redisplay.
0BFF: A4 D6    >342           ldy   selsave    ; -Yes, get selected index
0C01: BE C6 10 >343           ldx   fnxfn,y    ; -Yes, commit new
0C04: A0 0C    >344           ldy   #fnamecol  ;   file name.
0C06: C4 D8    >345  :copy    cpy   selch      ; End of file name?
0C08: F0 11    >346           beq   :fnend     ; -Yes.
0C0A: B1 DC    >347           lda   (selBASL),y
0C0C: 09 80    >348           ora   #$80       ; -No. Make normal.
0C0E: C9 A0    >349           cmp   #$A0       ; Upper case?
0C10: B0 02    >350           bcs   :norm      ; -No, already normal.
0C12: 09 40    >351           ora   #$40       ; -Yes, make normal.
0C14: 9D 00 11 >352  :norm    sta   fnames,x
0C17: E8       >353           inx
0C18: C8       >354           iny
0C19: D0 EB    >355           bne   :copy      ; (always)
               >356
```

```
0C1B: A9 00    >357  :fnend   lda   #0         ; Null at end
0C1D: 9D 00 11 >358           sta   fnames,x   ;  of fname.
0C20: A4 D6    >359           ldy   selsave    ; Reset Device Block
0C22: BE BE 10 >360           ldx   fnxdbx,y   ;  for new file.
0C25: 20 84 13 >361           jsr   resetdb
0C28: AD 00 C0 >362  :chkexit lda   KBD        ; Check last key.
0C2B: C9 1B    >363           cmp   #escape&$7F ; Was it ESCAPE?
0C2D: F0 03    >364           beq   :restart   ; -Yes, back to sim.
0C2F: 4C 76 0B >365  :disiocr jmp   disiocfg   ; Redisplay & continue.
               >366
0C32: 4C 47 09 >367  :restart jmp   restart    ; Restart B220SIM.
               >368
0C35: 84 D6    >369  :beep    sty   selsave    ; Scratch to save Y.
0C37: 20 DD FB >370           jsr   BEEP       ; Signal invalid key
0C3A: A4 D6    >371           ldy   selsave    ; Restore Y
0C3C: 4C DD 0B >372  :kbdlpr  jmp   :kbdloop   ;  and continue.
               >373
0C3F: C9 8A    >374  :notup   cmp   #dnarrow
0C41: F0 04    >375           beq   :down
0C43: C9 8D    >376           cmp   #$8D
0C45: D0 0A    >377           bne   :notdown   ; Not down arrow or return.
0C47: E6 D5    >378  :down    inc   selected   ; Move cursor down
0C49: A5 D5    >379           lda   selected   ;  and wrap around.
0C4B: 29 07    >380           and   #7
0C4D: 85 D5    >381           sta   selected
0C4F: 10 A4    >382           bpl   :edited    ; (always)
               >383
0C51: C9 9B    >384  :notdown cmp   #escape    ; ESC?
0C53: F0 A0    >385           beq   :edited    ; -Yes, commit fname.
0C55: C9 88    >386           cmp   #ltarrow   ; Left arrow?
0C57: F0 04    >387           beq   :backsp    ; -Yes, backspace.
0C59: C9 FF    >388           cmp   #delete    ; DELETE?
0C5B: D0 13    >389           bne   :addchar   ; -No, add character.
0C5D: C0 0C    >390  :backsp  cpy   #fnamecol  ; At start?
0C5F: F0 D4    >391           beq   :beep      ; -Yes, complain.
0C61: A9 A0    >392           lda   #"         " ; -No, blank cursor
0C63: 91 DC    >393           sta   (selBASL),y
0C65: 88       >394           dey              ; Back up.
0C66: A9 57    >395  :changed lda   #cursor    ; Place cursor.
0C68: 91 DC    >396           sta   (selBASL),y
0C6A: 84 D8    >397           sty   selch      ; Save cursor column.
0C6C: 85 DA    >398           sta   changed    ; Mark changed & cont.
0C6E: D0 CC    >399           bne   :kbdlpr    ; (always)
               >400
0C70: A6 DA    >401  :addchar ldx   changed    ; Any prior change?
0C72: D0 0D    >402           bne   :notfrst   ; -Yes, just add char.
0C74: AA       >403           tax              ; Save character.
0C75: A9 A0    >404           lda   #"         " ; Blank out file name.
0C77: C0 0C    >405  :cloop   cpy   #fnamecol
0C79: F0 05    >406           beq   :addit
0C7B: 91 DC    >407           sta   (selBASL),y
0C7D: 88       >408           dey
0C7E: D0 F7    >409           bne   :cloop     ; (always)
               >410
0C80: 8A       >411  :addit   txa              ; Restore character.
0C81: C0 24    >412  :notfrst cpy   #fnamecol+24 ; At end?
0C83: B0 B0    >413           bcs   :beep      ; -Yes, complain.
0C85: 20 8D 0C >414           jsr   inverse    ; -No, make inverse.
0C88: 91 DC    >415           sta   (selBASL),y ;  and add to fname.
0C8A: C8       >416           iny              ; Advance CH
0C8B: D0 D9    >417           bne   :changed   ; (always)
               >418
0C8D: 29 7F    >419  inverse  and   #$7F       ; Make inverse
0C8F: C9 40    >420           cmp   #$40       ; Upper case?
0C91: 90 06    >421           bcc   :rts       ; -No, special char.
0C93: C9 60    >422           cmp   #$60       ; Upper case?
0C95: B0 02    >423           bcs   :rts       ; -No, lower case.
```

===== Page 21 =====

```
0C97: 29 1F                  and   #$1F       ; -Yes, make inverse
0C99: 60            :rts     rts
```

```
         put   B220PANEL
**********************************************************
*                                                        *
*           B220 front panel display routines            *
*                                                        *
**********************************************************

off      equ   " "        ; blank (neon off)
on       equ   "*"        ; asterisk (neon on)

AR8      equ   $580       ; Line 4
AR4      equ   $600       ; Line 5
AR2      equ   $680       ; Line 6
AR1      equ   $700       ; Line 7
ARv      equ   $428       ; Line 9
BPC8     equ   $5A8       ; Line 12
BPC4     equ   $628       ; Line 13
BPC2     equ   $6A8       ; Line 14
BPC1     equ   $728       ; Line 15
BPCv     equ   $450       ; Line 17
STATlin  equ   $550       ; Line 19

B220col  equ   13-1       ; Leftmost title column
Acol     equ   6-1        ; Leftmost digit column of A
Rcol     equ   24-1       ; Leftmost digit column of R
Bcol     equ   6-1        ; Leftmost digit column of B
Pcol     equ   14-1       ; Leftmost digit column of P
Ccol     equ   22-1       ; Leftmost digit column of C
SW1col   equ   7-1        ; SW 1 column
RUNcol   equ   18-1       ; RUN column
ERRcol   equ   22-1       ; ERR column
COMPcol  equ   26-1       ; COMP column
OFLcol   equ   32-1       ; OFL column
RPTcol   equ   35-1       ; RPT column

* Register label addresses

Alab     equ   AR8+3
Rlab     equ   AR8+21
Blab     equ   BPC8+3
Plab     equ   BPC8+11
Clab     equ   BPC8+19
SWlab    equ   STATlin+3
ERRlab   equ   STATlin+ERRcol+2 ; Error type character
```

```
               >45   * Register front panel attributes
               >46
0C9A: 2D 04 05 >47   Aattr    dw    ARv+Acol,AR1+Acol,AR2+Acol,AR4+Acol,AR8+Acol
0CA4: A3       >48            db    rA+5       ; Low byte of rA
0CA5: 0B       >49            db    12-1       ; Display columns - 1
0CA6: 01 00 01 >50            db    1,0,1,1,1,1,1,1,1,1,1,1 ; Column mask
0CB2: 3F 04 17 >51   Rattr    dw    ARv+Rcol,AR1+Rcol,AR2+Rcol,AR4+Rcol,AR8+Rcol
0CBC: A9       >52            db    rR+5       ; Low byte of rR
0CBD: 0B       >53            db    12-1       ; Display columns - 1
0CBE: 01 00 01 >54            db    1,0,1,1,1,1,1,1,1,1,1,1 ; Column mask
0CCA: 55 04 2D >55   Battr    dw    BPCv+Bcol,BPC1+Bcol,BPC2+Bcol,BPC4+Bcol,BPC8+Bcol
0CD4: 95       >56            db    rB+1       ; Low byte of rB
0CD5: 03       >57            db    4-1        ; Display columns - 1
0CD6: 01 01 01 >58            db    1,1,1,1    ; Column mask
0CDA: 5D 04 35 >59   Pattr    dw    BPCv+Pcol,BPC1+Pcol,BPC2+Pcol,BPC4+Pcol,BPC8+Pcol
0CE4: 97       >60            db    rP+1       ; Low byte of rP
0CE5: 03       >61            db    4-1        ; Display columns - 1
0CE6: 01 01 01 >62            db    1,1,1,1    ; Column mask
0CEA: 65 04 3D >63   Cattr    dw    BPCv+Ccol,BPC1+Ccol,BPC2+Ccol,BPC4+Ccol,BPC8+Ccol
0CF4: 9D       >64            db    rC+5       ; Low byte of rC
0CF5: 0D       >65            db    14-1       ; Display columns - 1
0CF6: 01 00 01 >66            db    1,0,1,1,1,1,0,1,1,0,1,1,1,1 ; Column mask
```

```
                      ===== Page 24 =====

               >68   **********************************************************
               >69   *                                                        *
               >70   *              Initialize B220 Front Panel               *
               >71   *                                                        *
               >72   **********************************************************
               >73
0D04: D8       >74   disppanl cld             ; Force binary mode.
0D05: A9 15    >75            lda   #21        ; Disable 80-col firmware
0D07: 20 ED FD >76            jsr   COUT
0D0A: A9 00    >77            lda   #0
0D0C: 85 22    >78            sta   WNDTOP     ; Set full-screen window.
0D0E: 20 58 FC >79            jsr   HOME       ; Clear 40-col screen
0D11: 8D 0F C0 >80            sta   ALTCHAR    ; Select alternate charset
0D14: A2 0B    >81            ldx   #B220col-1
0D16: 20 4A F9 >82            jsr   PRBL2      ; Space to starting column
0D19: A0 00    >83            ldy   #0
0D1B: B9 BF 0D >84   :titloop lda   B220msg,y  ; Display title and AR top border
0D1E: F0 06    >85            beq   :AR
0D20: 20 ED FD >86            jsr   COUT
0D23: C8       >87            iny
0D24: D0 F5    >88            bne   :titloop   ; (always)
               >89
0D26: 20 95 0D >90   :AR      jsr   disARmid   ; Display 8-bit line
0D29: 20 95 0D >91            jsr   disARmid   ; Display 4-bit line
0D2C: 20 95 0D >92            jsr   disARmid   ; Display 2-bit line
0D2F: 20 95 0D >93            jsr   disARmid   ; Display 1-bit line
0D32: A0 00    >94            ldy   #0
0D34: B9 D4 0D >95   :ARborlp lda   ARbord,y   ; Display AR bottom border
0D37: F0 06    >96            beq   :BPC
0D39: 20 ED FD >97            jsr   COUT
0D3C: C8       >98            iny
0D3D: D0 F5    >99            bne   :ARborlp   ; (always)
               >100
0D3F: 20 8D 0D >101  :BPC     jsr   blanklin   ; <blank line for reg values>
0D42: 20 8D 0D >102           jsr   blanklin   ; <blank line>
0D45: 20 A3 0D >103           jsr   disBPCbo   ; Display BPC top border
0D48: 20 B1 0D >104           jsr   disBPCmi   ; Display 8-bit line
0D4B: 20 B1 0D >105           jsr   disBPCmi   ; Display 4-bit line
0D4E: 20 B1 0D >106           jsr   disBPCmi   ; Display 2-bit line
0D51: 20 B1 0D >107           jsr   disBPCmi   ; Display 1-bit line
0D54: 20 A3 0D >108           jsr   disBPCbo   ; Display BPC bottom border
0D57: 20 8D 0D >109           jsr   blanklin   ; <blank line for values>
0D5A: 20 8D 0D >110           jsr   blanklin   ; <blank line>
0D5D: A0 00    >111           ldy   #0         ; Display Status & Help lines
0D5F: B9 6C 0E >112  :STATlp  lda   STAT,y
0D62: F0 06    >113           beq   :finish
0D64: 20 ED FD >114           jsr   COUT
0D67: C8       >115           iny
0D68: D0 F5    >116           bne   :STATlp    ; (always)
               >117
0D6A: A9 81    >118  :finish  lda   #$81       ; "A" label
0D6C: 8D 83 05 >119           sta   Alab
0D6F: A9 82    >120           lda   #$82       ; "B" label
0D71: 8D AB 05 >121           sta   Blab
0D74: A9 83    >122           lda   #$83       ; "C" label
0D76: 8D BB 05 >123           sta   Clab
0D79: A9 90    >124           lda   #$90       ; "P" label
0D7B: 8D B3 05 >125           sta   Plab
0D7E: A9 92    >126           lda   #$92       ; "R" label
0D80: 8D 95 05 >127           sta   Rlab
0D83: A9 93    >128           lda   #$93       ; "S" of "Sw"
0D85: 8D 53 05 >129           sta   SWlab
0D88: A9 14    >130           lda   #20        ; Window is last 4 lines.
0D8A: 85 22    >131           sta   WNDTOP
0D8C: 60       >132           rts
               >133
0D8D: A9 A0    >134  blanklin lda   #"         " ; Separate CRs with blank
```

```
0D8F: 20 ED FD >135          jsr   COUT
0D92: 4C 8E FD >136          jmp   CROUT
               >137
0D95: A0 00    >138 disARmid ldy   #0         ; Display AR middle line
0D97: B9 FA 0D >139 :loop    lda   ARmid,y
0D9A: F0 06    >140          beq   :rts
0D9C: 20 ED FD >141          jsr   COUT
0D9F: C8       >142          iny
0DA0: D0 F5    >143          bne   :loop      ; (always)
               >144
0DA2: 60       >145 :rts     rts
               >146
0DA3: A0 00    >147 disBPCbo ldy   #0         ; Display BPC border
0DA5: B9 20 0E >148 :loop    lda   BPCbord,y
0DA8: F0 06    >149          beq   :rts
0DAA: 20 ED FD >150          jsr   COUT
0DAD: C8       >151          iny
0DAE: D0 F5    >152          bne   :loop      ; (always)
               >153
0DB0: 60       >154 :rts     rts
               >155
0DB1: A0 00    >156 disBPCmi ldy   #0         ; Display BPC middle line
0DB3: B9 46 0E >157 :loop    lda   BPCmid,y
0DB6: F0 06    >158          beq   :rts
0DB8: 20 ED FD >159          jsr   COUT
0DBB: C8       >160          iny
0DBC: D0 F5    >161          bne   :loop      ; (always)
               >162
0DBE: 60       >163 :rts     rts
               >164
0DBF: C2 F5 F2 >165 B220msg  asc   "Burroughs 220 v2.0"8DA08D
0DD4: A0 A0 A0 >166 ARbord   asc   "    +-+----------+    +-+----------+",8D00
0DFA: A0 A0 A0 >167 ARmid    asc   "    | |          |    | |          |",8D00
0E20: A0 A0 A0 >168 BPCbord  asc   "    +----+  +----+  +-+----+--+----+",8D00
0E46: A0 A0 A0 >169 BPCmid   asc   "    |    |  |    |  | |    |  |    |",8D00
0E6C: A0 A0 A0 >170 STAT     asc   "   Sw 0123456789 Run Err < = > Ov Rp",8DA08D
0E93: A0 D3 F4 >171 Help1    asc   " Stop/Step: <space>, Go: G, Reset: Z",8D
0EB8: A0 D3 E5 >172 Help2    asc   " Set reg: A/R/B/P/C + digits + Return",8D
0EDE: A0 D4 EF >173 Help3    asc   " Toggle switch: S + digit, Help: ?",8D
0F01: A0 C9 AF >174 Help4    asc   " I/O Config: I, Quit to BASIC: Q",00
               >175
0F22: 20 58 FC >176 disphelp jsr   HOME       ; Display help lines.
0F25: A0 00    >177          ldy   #0         ; (window is last 4 lines)
0F27: B9 93 0E >178 :helplp  lda   Help1,y
0F2A: F0 06    >179          beq   :done
0F2C: 20 ED FD >180          jsr   COUT
0F2F: C8       >181          iny
0F30: D0 F5    >182          bne   :helplp    ; (always)
               >183
0F32: 60       >184 :done    rts
```

```
                ***********************************************************
                *                                                         *
                *                   Display B220 State                    *
                *                                                         *
                ***********************************************************

0F33: 20 45 0F  display  jsr   dispA      ; Display A
0F36: 20 4C 0F           jsr   dispR      ; Display R
0F39: 20 53 0F           jsr   dispB      ; Display B
0F3C: 20 5A 0F           jsr   dispP      ; Display P
0F3F: 20 61 0F           jsr   dispC      ; Display C
0F42: 4C 68 0F           jmp   dispSTAT   ; Disp Status & return.

0F45: A9 9A     dispA    lda   #<Aattr    ; Register A attributes
0F47: A0 0C              ldy   #>Aattr
0F49: 4C F5 0F           jmp   dispreg    ; Display the register.

0F4C: A9 B2     dispR    lda   #<Rattr    ; Register R attributes
0F4E: A0 0C              ldy   #>Rattr
0F50: 4C F5 0F           jmp   dispreg    ; Display the register.

0F53: A9 CA     dispB    lda   #<Battr    ; Register B attributes
0F55: A0 0C              ldy   #>Battr
0F57: 4C F5 0F           jmp   dispreg    ; Display the register.

0F5A: A9 DA     dispP    lda   #<Pattr    ; Register P attributes
0F5C: A0 0C              ldy   #>Pattr
0F5E: 4C F5 0F           jmp   dispreg    ; Display the register.

0F61: A9 EA     dispC    lda   #<Cattr    ; Register C attributes
0F63: A0 0C              ldy   #>Cattr
0F65: 4C F5 0F           jmp   dispreg    ; Display the register.

0F68: A9 50     dispSTAT lda   #<STATlin  ; Set ptr to STATlin
0F6A: 85 CC              sta   ptr
0F6C: A9 05              lda   #>STATlin
0F6E: 85 CD              sta   ptr+1
0F70: A2 00              ldx   #0
0F72: A0 06              ldy   #SW1col    ; Start at switch 1
0F74: B5 B6     :swloop  lda   CSW,x      ; Is it on?
0F76: 20 CC 0F           jsr   INDshow    ; Display it's state
0F79: E8                 inx              ; Next switch
0F7A: E0 0A              cpx   #10        ; Until done...
0F7C: 90 F6              bcc   :swloop
0F7E: A0 11              ldy   #RUNcol
0F80: A5 C0              lda   RUN
0F82: 20 CC 0F           jsr   INDshow
0F85: A0 15              ldy   #ERRcol
0F87: A5 C1              lda   ERR
0F89: 20 CC 0F           jsr   INDshow
0F8C: A0 19              ldy   #COMPcol
0F8E: A5 C2              lda   COMP       ; <0, 0, >0: < = >
0F90: 30 07              bmi   :lt
0F92: F0 0A              beq   :eq
0F94: A2 0C              ldx   #:gtstr-:ltstr ; Point to > string
0F96: 4C A0 0F           jmp   :show

0F99: A2 00     :lt      ldx   #:ltstr-:ltstr ; Point to < string
0F9B: 4C A0 0F           jmp   :show

0F9E: A2 06     :eq      ldx   #:eqstr-:ltstr ; Point to = string
0FA0: BD BA 0F  :show    lda   :ltstr,x
0FA3: F0 06              beq   :next
0FA5: 91 CC              sta   (ptr),y
0FA7: C8                 iny
0FA8: E8                 inx
0FA9: D0 F5              bne   :show      ; (always)
```

```

0FAB: A0 1F    :next    ldy   #OFLcol
0FAD: A5 C3             lda   Ov         ; Overflow indicator
0FAF: 20 CC 0F          jsr   INDshow
0FB2: A0 22             ldy   #RPTcol
0FB4: A5 C4             lda   Rp         ; Repeat indicator
0FB6: 20 CC 0F          jsr   INDshow
0FB9: 60                rts

0FBA: 3C       :ltstr   asc   '<' ; Inverse
0FBB: A0 BD A0          asc   " = >",00
0FC0: BC A0    :eqstr   asc   "< "
0FC2: 3D                asc   '=' ; Inverse
0FC3: A0 BE 00          asc   " >",00
0FC6: BC A0 BD :gtstr   asc   "< = "
0FCA: 3E 00             asc   '>',00 ; inverse

               **********************************************************
               *                                                        *
               *     Flip indicator to on (inverse) or off (normal)     *
               *                                                        *
               * A = indicator: 0 is OFF, >0 is ON                      *
               * Y = leftmost column of indicator - 1                   *
               * Exits with Y pointing 1 past last column of indicator  *
               *                                                        *
               **********************************************************

0FCC: 18       INDshow  clc              ; >0 ==> inv, 0 ==> norm
0FCD: 69 FF             adc   #$FF       ; Set C if >0, reset if 0
0FCF: B1 CC    :loop    lda   (ptr),y    ; Get indicator char
0FD1: 29 20             and   #$20       ; Is it Upper Case?
0FD3: D0 06             bne   :notuc     ; -No, leave it alone.
0FD5: B1 CC             lda   (ptr),y    ; -Yes, turn off $40 bit
0FD7: 29 BF             and   #$BF       ;   to avoid mousetext.
0FD9: D0 02             bne   :switch    ; (always)

0FDB: B1 CC    :notuc   lda   (ptr),y    ; Recover original char
0FDD: 90 04    :switch  bcc   :norm      ; Set to normal
0FDF: 29 7F             and   #$7F       ; Set to inverse
0FE1: B0 02             bcs   :store     ; (always)

0FE3: 09 80    :norm    ora   #$80       ; Set to normal
0FE5: 91 CC    :store   sta   (ptr),y
0FE7: C8                iny              ; Advance to next char
0FE8: B1 CC             lda   (ptr),y    ;  and examine it.
0FEA: 09 80             ora   #$80       ; Force normal
0FEC: 49 A0             eor   #"         " ; Space?
0FEE: F0 04             beq   :done      ; -Yes, done.
0FF0: 29 E0             and   #$E0       ; -No, digit?
0FF2: D0 DB             bne   :loop      ; -No, keep going.
0FF4: 60       :done    rts              ; -Yes, done.
```

```
               **********************************************************
               *                                                        *
               *         Display a B220 register on front panel         *
               *                                                        *
               * Address of register attributes block is loaded in A,Y  *
               *                                                        *
               **********************************************************

0FF5: 85 CC    dispreg  sta   ptr        ; Set register attribute ptr
0FF7: 84 CD             sty   ptr+1
0FF9: A0 00             ldy   #0
0FFB: B1 CC    :cpyattr lda   (ptr),y    ; Copy reg attributes to page 0
0FFD: 99 D4 00          sta   linev,y
1000: C8                iny
1001: C0 0A             cpy   #10
1003: 90 F6             bcc   :cpyattr
1005: B1 CC             lda   (ptr),y    ; Addr of low byte of register
1007: 8D 1A 10          sta   :reg+1
100A: C8                iny
100B: B1 CC             lda   (ptr),y
100D: A8                tay              ; Set Y = rightmost column
100E: 18                clc
100F: A5 CC             lda   ptr        ; Advance ptr to digit mask
1011: 69 0C             adc   #12
1013: 85 CC             sta   ptr
1015: 90 02             bcc   :reg
1017: E6 CD             inc   ptr+1
1019: A5 00    :reg     lda   0*0        ; Load register byte
101B: CE 1A 10          dec   :reg+1     ;  and move to next highest.
101E: 85 D0             sta   t1         ; Save current reg byte
1020: 20 33 10          jsr   dispdig    ; Display lo digit of reg byte
1023: 88                dey              ; Move left one column.
1024: 30 0C             bmi   :done      ; Quit if done...
1026: 20 33 10          jsr   dispdig    ; Display hi digit of reg byte
1029: 88       :skip    dey              ; Move left.
102A: 30 06             bmi   :done      ; -Display complete.
102C: B1 CC             lda   (ptr),y    ; Check mask
102E: F0 F9             beq   :skip      ; -Skip this screen column
1030: D0 E7             bne   :reg       ; -Keep going...

1032: 60       :done    rts

```

```
               **********************************************************
               *                                                        *
               *           Display one digit of B220 register           *
               *                                                        *
               **********************************************************

1033: A5 D0    dispdig  lda   t1         ; Get (shifted) reg byte.
1035: 29 0F             and   #$0F       ; Mask low digit,
1037: 09 B0             ora   #$B0       ;  make ASCII digit,
1039: 91 D4             sta   (linev),y  ;   and store it on screen.
103B: 46 D0             lsr   t1         ; 1-bit to Carry
103D: A9 A0             lda   #off
103F: 90 02             bcc   :st1
1041: A9 AA             lda   #on
1043: 91 D6    :st1     sta   (line1),y  ; Store 1-bit state to screen
1045: 46 D0             lsr   t1         ; 2-bit to Carry
1047: A9 A0             lda   #off
1049: 90 02             bcc   :st2
104B: A9 AA             lda   #on
104D: 91 D8    :st2     sta   (line2),y  ; Store 2-bit state to screen
104F: 46 D0             lsr   t1         ; 4-bit to Carry
1051: A9 A0             lda   #off
1053: 90 02             bcc   :st4
1055: A9 AA             lda   #on
1057: 91 DA    :st4     sta   (line4),y  ; Store 4-bit state to screen
1059: 46 D0             lsr   t1         ; 8-bit to Carry
105B: A9 A0             lda   #off
105D: 90 02             bcc   :st8
105F: A9 AA             lda   #on
1061: 91 DC    :st8     sta   (line8),y  ; Store 8-bit state to screen
1063: 60                rts
```

```
                66          put   B220IO
               >1    ***********************************************************
               >2    *                                                         *
               >3    *               B220 Buffered I/O Routines                *
               >4    *                                                         *
               >5    ***********************************************************
               >6
               >7    * File/Buffer Parameters
               >8
               >9    fnlen    equ   25         ; File name max length
               >10   ptbfsz   equ   100*6      ; Paper tape buf: 100 words.
               >11   blksize  equ   101*6      ; block = Preface + 100 words.
               >12   mtbfsz   equ   10*blksize ; Mag Tape buf: 6060 bytes.
               >13
               >14   db       equ   *          ; Device Information Block
               >15
1064: 4C 3B    >16   bfstart  dw    ptrdr0bf   ; Paper tape reader 0 buffer
1066: 4C 3B    >17   bfptr    dw    ptrdr0bf   ; Current buf pointer
1068: A4 3D    >18   bfend    dw    ptrdr0bf+ptbfsz ; End of buffer + 1
106A: 58 02    >19   bfsiz    dw    ptbfsz     ; Buffer size in bytes
106C: 00       >20   bffn     db    0*fnlen    ; File name table index
106D: 00 00 00 >21   bfoff    db    0,0,0      ; bfstart file offset
1070: 00       >22   bflane   db    0          ; Mag tape lane = 0 or 1
1071: 00       >23   bfdirty  db    0          ; Buffer contents changed
               >24
               >25   dbsz     equ   *-db       ; DB size
               >26
1072: A6 3D    >27            dw    ptrdr1bf   ; Paper tape reader 1 buffer
1074: A6 3D    >28            dw    ptrdr1bf
1076: FE 3F    >29            dw    ptrdr1bf+ptbfsz
1078: 58 02    >30            dw    ptbfsz
107A: 19       >31            db    1*fnlen
107B: 00 00 00 >32            db    0,0,0
107E: 00       >33            db    0
107F: 00       >34            db    0
               >35
1080: 00 60    >36            dw    ptpch0bf   ; Paper tape punch 0 buffer
1082: 00 60    >37            dw    ptpch0bf
1084: 58 62    >38            dw    ptpch0bf+ptbfsz
1086: 58 02    >39            dw    ptbfsz
1088: 32       >40            db    2*fnlen
1089: 00 00 00 >41            db    0,0,0
108C: 00       >42            db    0
108D: 00       >43            db    0
               >44
108E: 5A 62    >45            dw    ptpch1bf   ; Paper tape punch 1 buffer
1090: 5A 62    >46            dw    ptpch1bf
1092: B2 64    >47            dw    ptpch1bf+ptbfsz
1094: 58 02    >48            dw    ptbfsz
1096: 4B       >49            db    3*fnlen
1097: 00 00 00 >50            db    0,0,0
109A: 00       >51            db    0
109B: 00       >52            db    0
               >53
109C: B4 64    >54            dw    mt0bf      ; Mag tape 0 buffer
109E: B4 64    >55            dw    mt0bf
10A0: 60 7C    >56            dw    mt0bf+mtbfsz
10A2: AC 17    >57            dw    mtbfsz
10A4: 64       >58            db    4*fnlen    ; (Lane 0)
10A5: 00 00 00 >59            db    0,0,0
10A8: 00       >60            db    0
10A9: 00       >61            db    0
               >62
10AA: 62 7C    >63            dw    mt1bf      ; Mag tape 1 buffer
10AC: 62 7C    >64            dw    mt1bf
10AE: 0E 94    >65            dw    mt1bf+mtbfsz
10B0: AC 17    >66            dw    mtbfsz
```

```
                              ===== Page 31 =====

10B2: 96       >67            db    6*fnlen    ; (Lane 0)
10B3: 00 00 00 >68            db    0,0,0
10B6: 00       >69            db    0
10B7: 00       >70            db    0
               >71
               >72   PTRclass equ   0          ; Paper Tape Reader class
               >73   PTPclass equ   2          ; Paper Tape Punch class
               >74   MTUclass equ   4          ; Mag Tape class
               >75
               >76   * Map Device Class + Unit ==> Device Block index
10B8: 00 0E 1C >77   classdbx db    0*dbsz,1*dbsz,2*dbsz
10BB: 2A 38 46 >78            db    3*dbsz,4*dbsz,5*dbsz
               >79
               >80   * Map filename index ==> Device Block index
10BE: 00 0E 1C >81   fnxdbx   db    0*dbsz,1*dbsz,2*dbsz,3*dbsz
10C2: 38 38 46 >82            db    4*dbsz,4*dbsz,5*dbsz,5*dbsz
               >83
               >84   * Map filename index ==> fn table index
10C6: 00 19 32 >85   fnxfn    db    0*fnlen,1*fnlen,2*fnlen,3*fnlen
10CA: 64 7D 96 >86            db    4*fnlen,5*fnlen,6*fnlen,7*fnlen
               >87
               >88   * I/O buffer definitions
               >89
               >90   ptrdr1bf equ   $4000-ptbfsz-2    ; Two PTRDR buffers
               >91   ptrdr0bf equ   ptrdr1bf-ptbfsz-2 ;  just below HGR2.
               >92
               >93            dum   $6000      ; Buffers in high Main mem
6000: 00 00 00 >94   ptpch0bf ds    ptbfsz+2
625A: 00 00 00 >95   ptpch1bf ds    ptbfsz+2
64B4: 00 00 00 >96   mt0bf    ds    mtbfsz+2
7C62: 00 00 00 >97   mt1bf    ds    mtbfsz+2
               >98            err   */$9600    ; Error if past $9600
               >99            dend
               >100
               >101  * File name table
               >102
               >103           align 256        ; Put table on page boundary
10CE: 00 00 00 >103           ds    *-1/256*256+256-*
               >103           eom
               >104
1100: D0 D4 D2 >105  fnames   asc   "PTRDR0",00
1107: 00 00 00 >106           ds    fnlen-7
1119: D0 D4 D2 >107           asc   "PTRDR1",00
1120: 00 00 00 >108           ds    fnlen-7
1132: D0 D4 D0 >109           asc   "PTPCH0",00
1139: 00 00 00 >110           ds    fnlen-7
114B: D0 D4 D0 >111           asc   "PTPCH1",00
1152: 00 00 00 >112           ds    fnlen-7
1164: CD D4 D5 >113           asc   "MTU0L0",00
116B: 00 00 00 >114           ds    fnlen-7
117D: CD D4 D5 >115           asc   "MTU0L1",00
1184: 00 00 00 >116           ds    fnlen-7
1196: CD D4 D5 >117           asc   "MTU1L0",00
119D: 00 00 00 >118           ds    fnlen-7
11AF: CD D4 D5 >119           asc   "MTU1L1",00
11B6: 00 00 00 >120           ds    fnlen-7
```

```
                >122  **********************************************************
                >123  *                                                        *
                >124  *              iosel - Select I/O device                 *
                >125  *                                                        *
                >126  * On entry: X = Device Class (0=RDR, 2=PCH, 4=MTP)       *
                >127  * On exit:  X = dbx = DB index, Y = 0, ptr = bfptr,      *
                >128  *           A = (ptr) = sign (flag) byte of next word.   *
                >129  *                                                        *
                >130  **********************************************************
                >131
11C8: A5 99     >132  iosel    lda   rC+sL      ; Get unit number.
11CA: 29 E0     >133           and   #$E0       ; Unit number > 0 or 1?
11CC: D0 61     >134           bne   ]IOerr1    ; -Yes, I/O error.
11CE: A5 99     >135           lda   rC+sL      ; Get unit number
11D0: 29 10     >136           and   #$10
11D2: F0 01     >137           beq   :zero      ; Unit 0
11D4: E8        >138           inx              ; Unit 1
11D5: BD B8 10 >139  :zero    lda   classdbx,x ; Map class + unit to DB index.
11D8: AA        >140           tax
11D9: 85 D2     >141           sta   dbx        ; Save DB index.
11DB: BD 66 10 >142  setptr   lda   bfptr,x
11DE: 85 CC     >143           sta   ptr        ; ptr = bfptr
11E0: BD 67 10 >144           lda   bfptr+1,x
11E3: 85 CD     >145           sta   ptr+1
11E5: A2 00     >146           ldx   #0
11E7: A1 CC     >147           lda   (ptr,x)    ; A = sign byte of next word.
11E9: A6 D2     >148           ldx   dbx        ; Restore X.
11EB: 60        >149           rts
                >150
                >151  **********************************************************
                >152  *                                                        *
                >153  *             iodsel - Deselect I/O device               *
                >154  *                                                        *
                >155  * On entry: dbx = DB index                               *
                >156  * On exit:  X = DB index, bfptr = ptr                    *
                >157  *                                                        *
                >158  **********************************************************
                >159
                >160
11EC: A6 D2     >161  iodsel   ldx   dbx        ; DB index.
11EE: A5 CC     >162           lda   ptr        ; bfptr = ptr.
11F0: 9D 66 10 >163           sta   bfptr,x
11F3: A5 CD     >164           lda   ptr+1
11F5: 9D 67 10 >165           sta   bfptr+1,x
11F8: 60        >166           rts
                >167
                >168  **********************************************************
                >169  *                                                        *
                >170  *       getwrd - Get next word from buffer into rD       *
                >171  *                                                        *
                >172  * On entry: ptr = pointer to next word in buffer,        *
                >173  *           dbx = DB index.                              *
                >174  * On exit:  rD = next word in buffer, ptr advanced.      *
                >175  *                                                        *
                >176  **********************************************************
                >177
11F9: A0 00     >178  getwrd   ldy   #0         ; Sign flag: EOF, EOB/Empty,
11FB: B1 CC     >179           lda   (ptr),y    ;  normal/Prefix?
11FD: C9 BA     >180  :again   cmp   #PREF+$A   ; Normal or prefix word?
11FF: B0 18     >181           bcs   :special   ; -No, EOF, EOB, or EMPTY.
1201: 85 AA     >182           sta   rD+S       ; -Yes, put sign in rD and
1203: A0 05     >183           ldy   #5         ;   copy rest of word to rD.
1205: B1 CC     >184  :getlp   lda   (ptr),y
1207: 99 AA 00 >185           sta   rD,y
120A: 88        >186           dey
120B: D0 F8     >187           bne   :getlp
120D: 18        >188  ]incptr6 clc              ; Increment ptr by 6.
```

```
120E: A5 CC              lda   ptr
1210: 69 06              adc   #$6
1212: 85 CC              sta   ptr
1214: 90 02              bcc   :rts
1216: E6 CD              inc   ptr+1
1218: 60        :rts     rts

1219: A6 D2     :special ldx   dbx        ; Point to Device Block.
121B: C9 EF              cmp   #EOF       ; End-Of-File?
121D: F0 10              beq   ]IOerr1    ; -Yes, I/O error.
121F: C9 EE              cmp   #EMPTY     ; -No. Is buffer empty?
1221: F0 06              beq   :load      ; -Yes, load buffer.
1223: 20 E3 13           jsr   flushbuf   ; -No, EOB. Flush buf to disk.
1226: 20 28 13           jsr   advoff     ; Advance buf offset.
1229: 20 73 12  :load    jsr   readbuf    ; Load the buffer
122C: 4C FD 11           jmp   :again     ;  and try again.

122F: 4C 21 08  ]IOerr1  jmp   X_IOerr    ; I/O error relay.

               **********************************************************
               *                                                        *
               *     putwrd - Put rD into next buffer word.             *
               *                                                        *
               * On entry: dbx = DB index, ptr current.                 *
               * On exit:  rD = next word in buffer, ptr advanced.      *
               *                                                        *
               **********************************************************

1232: A6 D2     putwrd   ldx   dbx        ; DB index.
1234: BD 68 10           lda   bfend,x    ; Is buffer full?
1237: C5 CC              cmp   ptr
1239: D0 15              bne   :notfull   ; -No, check empty.
123B: BD 69 10           lda   bfend+1,x
123E: C5 CD              cmp   ptr+1
1240: D0 0E              bne   :notfull   ; -No, check empty.
1242: 20 E3 13           jsr   flushbuf   ; -Yes, write if dirty,
1245: 20 28 13           jsr   advoff     ;   advance offset, and
1248: A9 EE              lda   #EMPTY     ;    mark buffer empty.
124A: A0 00              ldy   #0
124C: 91 CC              sta   (ptr),y
124E: F0 08              beq   :ckmtape   ; (always)

1250: A0 00     :notfull ldy   #0
1252: B1 CC              lda   (ptr),y
1254: C9 EE              cmp   #EMPTY     ; Is buffer empty?
1256: D0 0A              bne   :put       ; -No, put word.
1258: BD 6B 10  :ckmtape lda   bfsiz+1,x  ; -Yes, is device
125B: C9 17              cmp   #>mtbfsz       a mag tape?
125D: D0 03              bne   :put       ; -No. Put the word.
125F: 20 73 12           jsr   readbuf    ; -Yes, load the buffer.
1262: A9 01     :put     lda   #1         ; Mark buffer dirty.
1264: 9D 71 10           sta   bfdirty,x
1267: A0 05              ldy   #5         ; Move rD into buffer.
1269: B9 AA 00  :putlp   lda   rD,y
126C: 91 CC              sta   (ptr),y
126E: 88                 dey
126F: 10 F8              bpl   :putlp
1271: 30 9A              bmi   ]incptr6   ; Inc ptr & return. (always)

               **********************************************************
               *                                                        *
               *                      readbuf                           *
               *                                                        *
               * On entry: dbx = DB index.                              *
               * On exit:  X = dbx = DB index, Y = 0, ptr = bfstart,    *
               *           A = (ptr) = sign (flag) byte of next word.   *
               *                                                        *
               **********************************************************

1273: 20 92 13 readbuf  jsr   emptydb    ; Clear the buffer.
1276: 20 32 14          jsr   doread     ; Fill the buffer.
1279: A0 00             ldy   #0
127B: B1 CC             lda   (ptr),y    ; A = sign byte of next word.
127D: 60       ]rts     rts

               **********************************************************
               *                                                        *
               *      nxtblk - Advance ptr to point at next block.      *
               *                                                        *
               * On entry: X = DB index, A = (ptr) = sign flag.         *
               * On exit:  X unchanged, (ptr) = next block.             *
               *           I/O error if at EOF (unless op = MPE).       *
               *                                                        *
               **********************************************************

127E: 20 BC 12 nxtblk   jsr   ckpref     ; Position ptr at block preface.
1281: C9 EF    :nxt     cmp   #EOF       ; At End-Of-File?
1283: F0 14             beq   :ckmpe     ; -Yes, check for MPE.
1285: C9 EE             cmp   #EMPTY     ; -No. Is buffer empty?
1287: F0 0A             beq   :loadbf    ; -Yes, just load buffer.
1289: C9 EB             cmp   #EOB       ; -No. At End-Of-Buffer?
128B: D0 1D             bne   incblk     ; -No, just inc to next block.
128D: 20 E3 13          jsr   flushbuf   ; -Yes, flush the buffer,
1290: 20 28 13          jsr   advoff     ;        advance buf offset,
1293: 20 73 12 :loadbf  jsr   readbuf    ;         and fill the buffer.
1296: 4C 81 12          jmp   :nxt       ; Go again in fresh buffer.

1299: A5 9B    :ckmpe   lda   rC+OP      ; MPE opcode?
129B: C9 58             cmp   #$58
129D: D0 90             bne   ]IOerr1    ; -No, I/O error.
129F: A5 9A             lda   rC+VV      ; MPE variant?
12A1: 29 0F             and   #$0F
12A3: C9 02             cmp   #2
12A5: D0 88             bne   ]IOerr1    ; -No, I/O error.
12A7: B1 CC             lda   (ptr),y    ; -Yes, return with
12A9: 60                rts              ;        flag byte.

12AA: 18       incblk   clc              ; ptr = ptr + blksize.
12AB: A5 CC             lda   ptr
12AD: 69 5E             adc   #<blksize
12AF: 85 CC             sta   ptr
12B1: A5 CD             lda   ptr+1
12B3: 69 02             adc   #>blksize
12B5: 85 CD             sta   ptr+1
12B7: A0 00             ldy   #0
12B9: B1 CC             lda   (ptr),y    ; A = (ptr) = sign/flag byte.
12BB: 60                rts

12BC: A0 00    ckpref   ldy   #0         ; Position ptr to point
12BE: B1 CC    :ck      lda   (ptr),y    ;  at preface of current block.
12C0: C9 B0             cmp   #PREF
12C2: 90 01             bcc   :backup    ; If not there, back up.
12C4: 60                rts

12C5: 38       :backup  sec              ; ptr = ptr - 6.
12C6: A5 CC             lda   ptr
12C8: E9 06             sbc   #6
12CA: 85 CC             sta   ptr
12CC: B0 F0             bcs   :ck        ; No borrow. Check again.
12CE: C6 CD             dec   ptr+1
12D0: D0 EC             bne   :ck        ; Check again. (always)

               **********************************************************
               *                                                        *
               >323  *      prvblk - Adjust ptr to point at previous block.   *
               >324  *                                                        *
               >325  * On entry: X = DB index.                                *
               >326  * On exit:  X unchanged, A = (ptr) = next block, Y = 0.  *
               >327  *           I/O error if at beginning of file.           *
               >328  *                                                        *
               >329  **********************************************************
               >330
12D2: 20 BC 12 >331  prvblk   jsr   ckpref     ; Position ptr at block preface.
12D5: A5 CC    >332           lda   ptr        ; Is ptr at start of buffer?
12D7: DD 64 10 >333           cmp   bfstart,x
12DA: D0 1A    >334           bne   decblk     ; -No, just decrement ptr.
12DC: A5 CD    >335           lda   ptr+1
12DE: DD 65 10 >336           cmp   bfstart+1,x
12E1: D0 13    >337           bne   decblk     ; -No, just decrement ptr.
12E3: 20 E3 13 >338           jsr   flushbuf   ; -Yes, flush the buffer,
12E6: 20 08 13 >339           jsr   backoff    ;        back to prev buffer,
12E9: 20 73 12 >340           jsr   readbuf    ;         and fill the buffer.
12EC: BD 68 10 >341           lda   bfend,x    ; ptr = bfend.
12EF: 85 CC    >342           sta   ptr
12F1: BD 69 10 >343           lda   bfend+1,x
12F4: 85 CD    >344           sta   ptr+1
12F6: 38       >345  decblk   sec              ; ptr = ptr - blksize
12F7: A5 CC    >346           lda   ptr
12F9: E9 5E    >347           sbc   #<blksize
12FB: 85 CC    >348           sta   ptr
12FD: A5 CD    >349           lda   ptr+1
12FF: E9 02    >350           sbc   #>blksize
1301: 85 CD    >351           sta   ptr+1
1303: A0 00    >352           ldy   #0         ; A = (ptr) = sign/flag byte.
1305: B1 CC    >353           lda   (ptr),y
1307: 60       >354           rts
               >355
               >356  **********************************************************
               >357  *                                                        *
               >358  *      backoff - Back up bfoff by length of buffer.      *
               >359  *                                                        *
               >360  * On entry: X = DB index                                 *
               >361  * On exit:  X unchanged, bfoff backed up, ptr = bfstart. *
               >362  *           I/O error if offset goes below zero.         *
               >363  *                                                        *
               >364  **********************************************************
               >365
1308: 38       >366  backoff  sec              ; bfoff = bfoff - bfsiz.
1309: BD 6D 10 >367           lda   bfoff,x
130C: FD 6A 10 >368           sbc   bfsiz,x
130F: 9D 6D 10 >369           sta   bfoff,x
1312: BD 6E 10 >370           lda   bfoff+1,x
1315: FD 6B 10 >371           sbc   bfsiz+1,x
1318: 9D 6E 10 >372           sta   bfoff+1,x
131B: BD 6F 10 >373           lda   bfoff+2,x
131E: E9 00    >374           sbc   #0
1320: 9D 6F 10 >375           sta   bfoff+2,x
1323: 10 1B    >376           bpl   ]resptr    ; If +, set ptr = bfstart.
1325: 4C 21 08 >377           jmp   X_IOerr    ; Error if offset  0.
               >378
               >379  **********************************************************
               >380  *                                                        *
               >381  *      advoff - Advance bfoff by length of buffer.       *
               >382  *                                                        *
               >383  * On entry: X = DB index                                 *
               >384  * On exit:  X unchanged, bfoff advanced, ptr = bfstart.  *
               >385  *                                                        *
               >386  **********************************************************
               >387
1328: 18       >388  advoff   clc              ; bfoff = bfoff + bfsiz.
1329: BD 6D 10 >389           lda   bfoff,x
```

```
                              ===== Page 36 =====

132C: 7D 6A 10 >390          adc   bfsiz,x
132F: 9D 6D 10 >391          sta   bfoff,x
1332: BD 6E 10 >392          lda   bfoff+1,x
1335: 7D 6B 10 >393          adc   bfsiz+1,x
1338: 9D 6E 10 >394          sta   bfoff+1,x
133B: 90 03    >395          bcc   ]resptr
133D: FE 6F 10 >396          inc   bfoff+2,x
1340: BD 64 10 >397 ]resptr  lda   bfstart,x  ; ptr = bfstart.
1343: 85 CC    >398          sta   ptr
1345: BD 65 10 >399          lda   bfstart+1,x
1348: 85 CD    >400          sta   ptr+1
134A: 60       >401          rts
               >402
               >403 **********************************************************
               >404 *                                                        *
               >405 *                setlan - Set MTU lane                   *
               >406 *                                                        *
               >407 * On entry: X = dbx = DB index                           *
               >408 * On exit:  X unchanged, A = filename index              *
               >409 *                                                        *
               >410 **********************************************************
               >411
134B: A5 9A    >412 setlan   lda   rC+VV      ; Isolate lane #.
134D: 29 10    >413          and   #$10
134F: F0 02    >414          beq   :zero      ; Lane 0.
1351: A9 01    >415          lda   #1         ; Lane 1.
1353: DD 70 10 >416 :zero    cmp   bflane,x   ; Lane change?
1356: F0 1C    >417          beq   :done      ; -No, done.
1358: 48       >418          pha              ; -Yes, save new lane,
1359: 20 DB 11 >419          jsr   setptr     ;   ptr = bfptr(dbx).
135C: 20 E3 13 >420          jsr   flushbuf   ;    Flush current buffer,
135F: 20 92 13 >421          jsr   emptydb    ;     and set buffer empty.
1362: 68       >422          pla
1363: 9D 70 10 >423          sta   bflane,x   ; Set new lane
1366: A8       >424          tay              ; Compute new filname index.
1367: EC BC 10 >425          cpx   classdbx+4 ; Mag Tape unit 0 or 1?
136A: F0 02    >426          beq   :unit0     ; -Unit 0 ==> fnx = 4 + lane
136C: C8       >427          iny              ; -Unit 1 ==> fnx = 6 + lane
136D: C8       >428          iny
136E: B9 CA 10 >429 :unit0   lda   fnxfn+4,y  ; Get new lane filename
1371: 9D 6C 10 >430          sta   bffn,x     ;  index and save it.
1374: 60       >431 :done    rts
               >432
               >433 **********************************************************
               >434 *                                                        *
               >435 *                      resetdbs                          *
               >436 *                                                        *
               >437 **********************************************************
               >438
1375: A0 05    >439 resetdbs ldy   #ndb-1     ; Reset all Devices
1377: BE B8 10 >440 :resetlp ldx   classdbx,y
137A: 20 DB 11 >441          jsr   setptr     ; ptr = bfptr(dbx).
137D: 20 84 13 >442          jsr   resetdb
1380: 88       >443          dey
1381: 10 F4    >444          bpl   :resetlp
1383: 60       >445          rts
               >446
               >447 **********************************************************
               >448 *                                                        *
               >449 *                      resetdb                           *
               >450 *                                                        *
               >451 * On entry: X = DB index                                 *
               >452 * On exit:  dbx = X = DB index, Y unchanged,             *
               >453 *           Buffer cleared and set to EMPTY.             *
               >454 *                                                        *
               >455 **********************************************************
               >456
```

```
1384: 20 E3 13 >457  resetdb  jsr   flushbuf   ; Flush buffer.
1387: A9 00    >458           lda   #0
1389: 9D 6D 10 >459           sta   bfoff,x    ; Set offset = 0
138C: 9D 6E 10 >460           sta   bfoff+1,x
138F: 9D 6F 10 >461           sta   bfoff+2,x
1392: BD 64 10 >462  emptydb  lda   bfstart,x  ; ptr = bfptr = bfstart.
1395: 9D 66 10 >463           sta   bfptr,x
1398: 85 CC    >464           sta   ptr
139A: BD 65 10 >465           lda   bfstart+1,x
139D: 9D 67 10 >466           sta   bfptr+1,x
13A0: 85 CD    >467           sta   ptr+1
13A2: 98       >468           tya              ; Save Y.
13A3: 48       >469           pha
13A4: A0 00    >470           ldy   #0
13A6: A9 EE    >471           lda   #EMPTY     ; Mark buffer empty
13A8: D0 02    >472           bne   :store     ; Store EMPTY flag. (always)
               >473
13AA: A9 00    >474  :clearlp lda   #0         ; Clear buffer flag bytes.
13AC: 91 CC    >475  :store   sta   (ptr),y    ; Store flag byte.
13AE: 20 0D 12 >476           jsr   ]incptr6
13B1: A5 CD    >477           lda   ptr+1      ; At end of buffer?
13B3: DD 69 10 >478           cmp   bfend+1,x
13B6: 90 F2    >479           bcc   :clearlp   ; -No, keep clearing flags.
13B8: A5 CC    >480           lda   ptr
13BA: DD 68 10 >481           cmp   bfend,x
13BD: D0 EB    >482           bne   :clearlp
13BF: A9 EB    >483           lda   #EOB       ; -Yes, set End-Of-Buffer
13C1: 91 CC    >484           sta   (ptr),y    ;   after final block.
13C3: BD 64 10 >485           lda   bfstart,x  ; ptr = bfstart.
13C6: 85 CC    >486           sta   ptr
13C8: BD 65 10 >487           lda   bfstart+1,x
13CB: 85 CD    >488           sta   ptr+1
13CD: 68       >489           pla              ; Restore Y.
13CE: A8       >490           tay
13CF: 60       >491           rts
               >492
               >493  ***********************************************************
               >494  *                                                         *
               >495  *                      flushall                           *
               >496  *                                                         *
               >497  ***********************************************************
               >498
13D0: A0 05    >499  flushall ldy   #ndb-1     ; Flush all but PTR buffers.
13D2: BE B8 10 >500  :flushlp ldx   classdbx,y ; DB index
13D5: 86 D2    >501           stx   dbx        ; Set dbx.
13D7: 20 DB 11 >502           jsr   setptr     ; ptr = bfptr(dbx)
13DA: 20 E3 13 >503           jsr   flushbuf   ; Flush a buffer
13DD: 88       >504           dey
13DE: C0 01    >505           cpy   #1         ; Go until PTR buffers
13E0: D0 F0    >506           bne   :flushlp   ;  (1 and 0) are reached.
13E2: 60       >507           rts
               >508
               >509  ***********************************************************
               >510  *                                                         *
               >511  *                      flushbuf                           *
               >512  *                                                         *
               >513  * On entry: X = DB index                                  *
               >514  * On exit:  Buffer clean, ptr, bfptr, bfoff unchanged.    *
               >515  *           X,Y unchanged, A scrambled, dbx = DB index.   *
               >516  *                                                         *
               >517  ***********************************************************
               >518
13E3: 86 D2    >519  flushbuf stx   dbx        ; Set Device Block index.
13E5: BD 71 10 >520           lda   bfdirty,x  ; Does buf need to be written?
13E8: F0 09    >521           beq   :clean     ; -No, it's clean.
13EA: 98       >522           tya              ; -Yes, save Y
13EB: 48       >523           pha
```

```
13EC: 20 F4 13 >524           jsr   dowrite    ;   and do it...
13EF: 68       >525           pla              ; Restore Y.
13F0: A8       >526           tay
13F1: A6 D2    >527           ldx   dbx        ; Restore X.
13F3: 60       >528  :clean   rts
               >529
               >530  **********************************************************
               >531  *                                                        *
               >532  *                      dowrite                           *
               >533  *                                                        *
               >534  * On entry: dbx = DB index, ptr = current                *
               >535  * On exit:  X = dbx, bfptr = ptr (unchanged), buf clean. *
               >536  *                                                        *
               >537  **********************************************************
               >538
13F4: A6 D2    >539  dowrite  ldx   dbx        ; Get DB index.
13F6: 20 EC 11 >540           jsr   iodsel     ; Save 'ptr' in 'bfptr'.
13F9: A9 14    >541           lda   #>bsave    ; Set for write
13FB: A0 DF    >542           ldy   #<bsave
13FD: 20 72 14 >543           jsr   PDfae      ; "BSAVE <fn>,A$<bfstart>,E$"
1400: 4C 06 14 >544           jmp   :ckeof     ; Are we at End-Of-File?
               >545
1403: 20 AA 12 >546  :findlp  jsr   incblk     ; Advance to next block.
1406: A0 00    >547  :ckeof   ldy   #0         ; Check prefix sign/flag byte.
1408: B1 CC    >548           lda   (ptr),y
140A: C9 B0    >549           cmp   #PREF      ; Is ptr at block start?
140C: 90 21    >550           bcc   ]IOerr2    ; -No, block sync error.
140E: C9 EF    >551           cmp   #EOF       ; -Yes, are we at End-Of-File?
1410: F0 05    >552           beq   :useptr    ; -Yes, write EOF to file.
1412: C9 EB    >553           cmp   #EOB       ; -No, are we at End-Of-Buffer?
1414: D0 ED    >554           bne   :findlp    ; -No, search forward by block.
1416: 18       >555           clc              ; -Yes, don't write EOB.
1417: A5 CC    >556  :useptr  lda   ptr        ; If not C, use ptr - 1.
1419: E9 00    >557           sbc   #0         ; If C, just use ptr.
141B: A8       >558           tay
141C: A5 CD    >559           lda   ptr+1
141E: E9 00    >560           sbc   #0
1420: 20 AA 14 >561           jsr   PDebx      ; "<ptr>,B$<off>", Execute.
1423: B0 0A    >562           bcs   ]IOerr2
1425: A6 D2    >563           ldx   dbx
1427: A9 00    >564           lda   #0
1429: 9D 71 10 >565           sta   bfdirty,x  ; Mark buffer clean.
142C: 4C DB 11 >566           jmp   setptr     ; Restore ptr and return.
               >567
142F: 4C 21 08 >568  ]IOerr2  jmp   X_IOerr    ; I/O error.
               >569
               >570  **********************************************************
               >571  *                                                        *
               >572  *                       doread                           *
               >573  *                                                        *
               >574  * On entry: dbx = DB index, ptr = current                *
               >575  * On exit:  A = 0, X = dbx, ptr = bfstart, buffer clean. *
               >576  *                                                        *
               >577  **********************************************************
               >578
1432: A9 14    >579  doread   lda   #>bload    ; Set for read.
1434: A0 D8    >580           ldy   #<bload
1436: 20 72 14 >581           jsr   PDfae      ; "BLOAD <fn>,A$<start>,E$"
1439: BC 68 10 >582           ldy   bfend,x    ; E param is bfend.
143C: BD 69 10 >583           lda   bfend+1,x
143F: 20 AA 14 >584           jsr   PDebx      ; "<end>,B$<off>", Execute.
1442: A6 D2    >585           ldx   dbx        ; Load DB index.
1444: 90 0F    >586           bcc   :noerr     ; No error.
1446: 29 FE    >587           and   #$FE       ; Fold error 6 & 7 together.
1448: C9 06    >588           cmp   #6         ; "Path Not Found" error?
144A: D0 E3    >589           bne   ]IOerr2    ; -No, IOerr.
144C: 20 40 13 >590           jsr   ]resptr    ; -Yes, set 'ptr' to 'bfstart'
```

```
144F: A9 EF             lda   #EOF       ;   and set End-Of-File.
1451: A0 00             ldy   #0
1453: 91 CC             sta   (ptr),y
1455: A0 00    :noerr   ldy   #0
1457: 98                tya
1458: 9D 71 10          sta   bfdirty,x  ; Mark buffer clean.
145B: BD 68 10          lda   bfend,x    ; ptr = bfend.
145E: 85 CC             sta   ptr
1460: BD 69 10          lda   bfend+1,x
1463: 85 CD             sta   ptr+1
1465: B1 CC             lda   (ptr),y
1467: C9 EF             cmp   #EOF       ; (bfend) = End-Of-File?
1469: F0 04             beq   :done      ; -Yes, done.
146B: A9 EB             lda   #EOB       ; -No, set End-Of-Buffer
146D: 91 CC             sta   (ptr),y    ;       in (bfend).
146F: 4C 40 13 :done    jmp   ]resptr    ;        reset ptr to bfstart.

               **********************************************************
               *                                                        *
               *                  PDfae / PDebx                         *
               *                                                        *
               * On entry: dbx = DB index, ptr = current                *
               * On exit:  X = dbx, ptr unchanged.                      *
               *                                                        *
               **********************************************************

               zeroff   equ   line1      ; Zero offset flag

1472: A2 00    PDfae    ldx   #0         ; Start ProDOS command.
1474: 20 0E 15          jsr   putpdcmd   ; BLOAD or BSAVE.
1477: A4 D2             ldy   dbx        ; Y = Device Block index.
1479: B9 6B 10          lda   bfsiz+1,y  ; Init 'zeroff' to 0 to
147C: 49 02             eor   #>ptbfsz   ;  skip B param if PT unit
147E: 85 D6             sta   zeroff     ;   and offset = 0.
1480: B9 6C 10          lda   bffn,y     ; (A,Y) --> file name
1483: A8                tay
1484: A9 11             lda   #>fnames
1486: 20 0E 15          jsr   putpdcmd   ; Add file name.
1489: A9 14             lda   #>Aparm
148B: A0 E6             ldy   #<Aparm
148D: 20 0E 15          jsr   putpdcmd   ; Add ",A$".
1490: A4 D2             ldy   dbx
1492: B9 65 10          lda   bfstart+1,y ; address = bfstart
1495: 48                pha
1496: B9 64 10          lda   bfstart,y
1499: A8                tay
149A: 68                pla
149B: 20 F2 14          jsr   putwdhx    ; Add hex address...
149E: A9 14             lda   #>Eparm
14A0: A0 EA             ldy   #<Eparm
14A2: 20 0E 15          jsr   putpdcmd   ; Add ",E$"
14A5: 86 D7             stx   savex      ; Save ProDOS cmd index.
14A7: A6 D2             ldx   dbx
14A9: 60                rts

14AA: A6 D7    PDebx    ldx   savex      ; Restore command index.
14AC: 20 F2 14          jsr   putwdhx    ; Add length
14AF: 86 D7             stx   savex      ; Save X before "B" param
14B1: A9 14             lda   #>Bparm
14B3: A0 EE             ldy   #<Bparm
14B5: 20 0E 15          jsr   putpdcmd   ; Add ",B$"
14B8: A9 03             lda   #3         ; Offset has 3 bytes.
14BA: 85 CE             sta   inptr
14BC: A4 D2             ldy   dbx
14BE: C8                iny              ; Adjust dbx for bfoff+2
14BF: C8                iny
14C0: B9 6D 10 :offlp   lda   bfoff,y    ; MSB of offset first.
```

```
14C3: F0 02    >658            beq   :zero
14C5: 85 D6    >659            sta   zeroff     ; Remember non-zero offset.
14C7: 20 F6 14 >660  :zero     jsr   putbyte    ; Add next offset byte.
14CA: 88       >661            dey              ; Next-most-sig offset byte.
14CB: C6 CE    >662            dec   inptr      ; More offset bytes?
14CD: D0 F1    >663            bne   :offlp     ; -Yes, continue.
14CF: A5 D6    >664            lda   zeroff     ; -No. Is offset zero?
14D1: D0 02    >665            bne   :useB      ; -No, existing file, use B.
14D3: A6 D7    >666            ldx   savex      ; -Yes, new file, no B.
14D5: 4C 25 15 >667  :useB     jmp   pdosxeq    ; Execute command and return.
               >668
14D8: C2 CC CF >669  bload     asc   "BLOAD ",00
14DF: C2 D3 C1 >670  bsave     asc   "BSAVE ",00
14E6: AC C1 A4 >671  Aparm     asc   ",A$",00
14EA: AC C5 A4 >672  Eparm     asc   ",E$",00
14EE: AC C2 A4 >673  Bparm     asc   ",B$",00
               >674
14F2: 20 F6 14 >675  putwdhx   jsr   putbyte    ; Put first byte in hex
14F5: 98       >676            tya              ;  and fall into putbyte.
14F6: 48       >677  putbyte   pha              ; Save byte
14F7: 4A       >678            lsr
14F8: 4A       >679            lsr
14F9: 4A       >680            lsr
14FA: 4A       >681            lsr
14FB: 20 FF 14 >682            jsr   :stdig     ; Put hi hex digit
14FE: 68       >683            pla              ;  and then lo dig.
14FF: 29 0F    >684  :stdig    and   #$0F       ; Isolate digit
1501: 09 B0    >685            ora   #"0"       ; Or in zone
1503: C9 BA    >686            cmp   #$BA       ; >9?
1505: 90 02    >687            bcc   :store     ; -No, store it.
1507: 69 06    >688            adc   #6         ; -Yes, cvt to A..F
1509: 9D 00 02 >689  :store    sta   IN,x       ; Add char to IN buffer.
150C: E8       >690            inx
150D: 60       >691            rts
```

```
                        put   B220PDOS
               **********************************************************
               *                                                        *
               *                      PUTPDCMD                          *
               *                                                        *
               * Append null-terminated string at (A,Y) onto IN,X.      *
               * Command is in hi-ASCII.  A is hi, Y is lo.             *
               *                                                        *
               * Advances X, destroys A, Y, and 'inptr'.                *
               *                                                        *
               **********************************************************

150E: 85 CF    putpdcmd sta   inptr+1    ; Set up string pointer
1510: 84 CE             sty   inptr
1512: A0 00             ldy   #0
1514: B1 CE    :cmdloop lda   (inptr),y  ; Append command string
1516: F0 07             beq   :rts       ;  until null
1518: 9D 00 02          sta   IN,x       ;   to keyboard buffer.
151B: E8                inx              ; Bump pointers.
151C: C8                iny
151D: D0 F5             bne   :cmdloop   ; (always)

151F: 60       :rts     rts              ; Return...

               **********************************************************
               *                                                        *
               *                      PDOSCMD                           *
               *                                                        *
               * Execute null-terminated ProDOS command at (A,Y)        *
               * Command is in hi-ASCII.                                *
               *                                                        *
               * Keyboard buffer, sptr, and Y are changed.              *
               * On error, C is set and A contains error code.          *
               *                                                        *
               **********************************************************

1520: A2 00    pdoscmd  ldx   #0         ; Empty kbd buffer.
1522: 20 0E 15          jsr   putpdcmd   ; Move in the command
                                         ;  and fall into pdosxeq.

               **********************************************************
               *                                                        *
               *                      PDOSXEQ                           *
               *                                                        *
               * Execute ProDOS command in keyboard buffer after        *
               * appending a carriage return.  Command is in hi-ASCII.  *
               *                                                        *
               * On error, C is set and A contains error code.          *
               *                                                        *
               **********************************************************

1525: A9 8D    pdosxeq  lda   #$8D       ; Carriage Return
1527: 9D 00 02          sta   IN,x       ;  at end
152A: AD 42 BE          lda   BSSTATE    ; Save BASIC.SYSTEM
152D: 48                pha              ;  'state' var & set it
152E: A9 FF             lda   #$FF       ;    to suppress blank
1530: 8D 42 BE          sta   BSSTATE    ;     line.
1533: 20 03 BE          jsr   DOSCMD     ; Then do it...
1536: AA                tax              ; Save error code.
1537: 68                pla              ; Restore BASIC.SYSTEM
1538: 8D 42 BE          sta   BSSTATE    ;  state variable.
153B: 8A                txa              ; A = ProDOS error code.
153C: 60                rts
```

```
                 68           err   */ptrdr0bf ; Can't overrun buffers.
                 69
                 70   AUXcode  equ   *          ; Start of Aux code
                 71            org   endcomm    ; Aux mem origin
                 72            put   B220FETCH
                >1    **********************************************************
                >2    *                                                        *
                >3    *            Simulate next B220 Instruction              *
                >4    *                                                        *
                >5    **********************************************************
                >6
08D7: 4C CF 09 >7    ADDRerrR jmp   ADDRerr    ; Relay branch
08DA: 4C D9 09 >8    UNDIGerR jmp   UNDIGerr   ; Relay branch
08DD: 4C 3A 08 >9    keyinR   jmp   M_keyin    ; Relay branch to Main
08E0: 4C 43 08 >10   stopR    jmp   M_stop     ; Relay branch to Main
               >11
               >12   * Convert rP to instruction address
               >13
08E3: A6 97    >14   newP     ldx   rP+1       ; Low 2 BCD digits of rP
08E5: E0 9A    >15            cpx   #$99+1     ; Undigits?
08E7: B0 F1    >16            bcs   UNDIGerR   ; -Yes, error.
08E9: A4 96    >17            ldy   rP         ; High 2 BCD digits of rP
08EB: C0 4A    >18            cpy   #$49+1     ; ADDR error?
08ED: B0 E8    >19            bcs   ADDRerrR   ; -Yes, stop.
08EF: BD C7 19 >20            lda   BCDLadrl,x ; -No, compute 'instptr'
08F2: 79 FB 1A >21            adc   BCDHadrl,y
08F5: 85 C8    >22            sta   instptr    ; Low byte of instr address
08F7: BD 61 1A >23            lda   BCDLadrh,x
08FA: 79 45 1B >24            adc   BCDHadrh,y
08FD: B0 DB    >25            bcs   UNDIGerR   ; Carry out ==> undigit(s)
08FF: 85 C9    >26            sta   instptr+1  ; High byte of instr address
0901: A0 00    >27   fetch    ldy   #0         ; Fetch next instruction.
0903: 84 C6    >28            sty   skipincP   ; Don't skip incP
0905: B1 C8    >29            lda   (instptr),y
0907: 85 98    >30            sta   rC+S       ; Sign
0909: C8       >31            iny
090A: B1 C8    >32            lda   (instptr),y
090C: 85 99    >33            sta   rC+sL      ; (field) start, Length
090E: C8       >34            iny
090F: B1 C8    >35            lda   (instptr),y
0911: 85 9A    >36            sta   rC+VV      ; Variants
0913: C8       >37            iny
0914: B1 C8    >38            lda   (instptr),y
0916: 85 9B    >39            sta   rC+OP      ; OPcode
0918: C8       >40            iny
0919: B1 C8    >41            lda   (instptr),y
091B: 85 9C    >42            sta   rC+ADDR    ; High 2 digits of ADDR
091D: C8       >43            iny
091E: B1 C8    >44            lda   (instptr),y
0920: 85 9D    >45            sta   rC+ADDR+1  ; Low 2 digits of ADDR
0922: A5 98    >46   execute  lda   rC+S       ; Is Sign negative?
0924: 29 01    >47            and   #1
0926: F0 0F    >48            beq   :noBmod    ; -No, skip rB modification
0928: F8       >49            sed              ; / Decimal mode
0929: 18       >50            clc
092A: A5 9D    >51            lda   rC+ADDR+1  ; Add rB to rC+ADDR
092C: 65 95    >52            adc   rB+1
092E: 85 9D    >53            sta   rC+ADDR+1
0930: A5 9C    >54            lda   rC+ADDR
0932: 65 94    >55            adc   rB
0934: 85 9C    >56            sta   rC+ADDR
0936: D8       >57            cld              ; \ Back to binary mode
0937: AD 00 C0 >58   :noBmod  lda   KBD        ; User interaction?
093A: 30 A1    >59            bmi   keyinR     ; -Yes, handle it.
093C: A5 C0    >60            lda   RUN        ; RUN mode off
093E: 25 9B    >61            and   rC+OP      ;  or HLT instruction?
0940: F0 9E    >62            beq   stopR      ; -Yes, stop.
```

```
0942: 8D 30 C0 >63            sta   SPKR       ; -No, toggle speaker.
0945: C6 D3    >64            dec   dispctr    ; Update display every
0947: 10 07    >65            bpl   ]contin    ;  'dispcnt' instructions.
0949: A9 64    >66            lda   #dispcnt   ; Reset counter
094B: 85 D3    >67            sta   dispctr
094D: 20 4C 08 >68            jsr   M_disp
0950: A4 9B    >69   ]contin  ldy   rC+OP      ; Op code
0952: C0 60    >70            cpy   #$60       ; OP out of range?
0954: B0 6D    >71            bcs   OPerr      ; -Yes, stop.
0956: A5 C3    >72            lda   Ov         ; -No, is Overflow set
0958: 25 C7    >73            and   OvHlt      ;   and Ovflo Halt mode?
095A: F0 04    >74            beq   :ok        ; -No, continue.
095C: C0 31    >75            cpy   #$31       ; -Yes, is OP BOF?
095E: D0 67    >76            bne   OFLerr     ; -No, Overflow error.
0960: A5 C6    >77   :ok      lda   skipincP   ; -Yes, skip increment P?
0962: D0 03    >78            bne   :skip      ; -Yes, PRB hit sign 6/7.
0964: 20 A3 09 >79            jsr   incP       ; -No, inc rP and instptr.
0967: B9 EC 09 >80   :skip    lda   optabl,y   ; Get execute address.
096A: 8D 9A 09 >81            sta   :go+1
096D: B9 46 0A >82            lda   optabh,y   ; High bit set?
0970: 30 2A    >83            bmi   :noADDR    ; -Yes, ignore ADDR
0972: 8D 9B 09 >84            sta   :go+2      ; -No, save execute address
0975: A6 9D    >85            ldx   rC+ADDR+1  ; Low 2 BCD ADDR digits
0977: E0 9A    >86            cpx   #$99+1     ; Undigits?
0979: B0 5E    >87            bcs   UNDIGerr   ; -Yes, error.
097B: A4 9C    >88            ldy   rC+ADDR    ; High 2 BCD ADDR digits
097D: C0 4A    >89            cpy   #$49+1     ; ADDR error?
097F: B0 4E    >90            bcs   ADDRerr    ; -Yes, stop.
0981: BD C7 19 >91            lda   BCDLadrl,x ; -No, compute 'memptr'
0984: 79 FB 1A >92            adc   BCDHadrl,y
0987: 85 CA    >93            sta   memptr     ; Low byte of memory address
0989: BD 61 1A >94            lda   BCDLadrh,x
098C: 79 45 1B >95            adc   BCDHadrh,y
098F: B0 48    >96            bcs   UNDIGerr   ; Carry out ==> undigit(s).
0991: 85 CB    >97            sta   memptr+1   ; High byte of memory address
0993: A0 00    >98            ldy   #0         ; Enter execute with Y=0
0995: B1 CA    >99            lda   (memptr),y ;  & operand sign in A & rD+S.
0997: 85 AA    >100           sta   rD+S
0999: 4C 00 00 >101  :go      jmp   0*0        ; Go to execute routine.
               >102
099C: 29 7F    >103  :noADDR  and   #$7F       ; Turn off "noADDR" bit
099E: 8D 9B 09 >104           sta   :go+2      ;  and save execute address.
09A1: D0 F6    >105           bne   :go        ; (always)
               >106
               >107  * Increment rP and instptr
               >108
09A3: F8       >109  incP     sed              ; / BCD mode arithmetic
09A4: 18       >110           clc
09A5: A5 97    >111           lda   rP+1       ; Increment rP by 1
09A7: 69 01    >112           adc   #1
09A9: 85 97    >113           sta   rP+1
09AB: 90 0A    >114           bcc   :nocar     ; Hi digits don't change.
09AD: A5 96    >115           lda   rP         ; Propagate carry.
09AF: 69 00    >116           adc   #0
09B1: 85 96    >117           sta   rP
09B3: C9 4A    >118           cmp   #$49+1     ; Did we pass 4999?
09B5: B0 18    >119           bcs   ADDRerr    ; -Yes, ADDR error.
09B7: D8       >120  :nocar   cld              ; \ Back to binary.
09B8: A5 C8    >121           lda   instptr    ; Inc 'instptr' by 6
09BA: 69 06    >122           adc   #6
09BC: 85 C8    >123           sta   instptr
09BE: 90 02    >124           bcc   :nocarry
09C0: E6 C9    >125           inc   instptr+1
09C2: 60       >126  :nocarry rts
```

```
              >128  * B220 error routines
              >129
09C3: A9 CF   >130  OPerr    lda   #"O"       ; OPcode error
09C5: D0 14   >131           bne   ]err       ; (always)
              >132
09C7: A9 D6   >133  OFLerr   lda   #"V"       ; Overflow error
09C9: D0 10   >134           bne   ]err       ; (always)
              >135
09CB: A9 C6   >136  FIELDerr lda   #"F"       ; Field error
09CD: D0 0C   >137           bne   ]err       ; (always)
              >138
09CF: A9 C1   >139  ADDRerr  lda   #"A"       ; Address error
09D1: D0 08   >140           bne   ]err       ; (always)
              >141
09D3: 85 00   >142  IOerr    sta   0          ; Save I/O err code
09D5: A9 C9   >143           lda   #"I"       ; I/O error
09D7: D0 02   >144           bne   ]err
              >145
09D9: A9 D8   >146  UNDIGerr lda   #"X"       ; Non-BCD digit error
09DB: 8D 04 C0 >147 ]err     sta   WRITMAIN   ; Store to text screen
09DE: 8D 67 05 >148          sta   ERRlab     ; Show on screen.
09E1: 8D 05 C0 >149          sta   WRITAUX    ; Back to Auxmem
09E4: 85 C1   >150           sta   ERR        ; Set error indicator,
09E6: 20 DD FB >151          jsr   BEEP       ;  sound beep,
09E9: 4C 43 08 >152          jmp   M_stop     ;   and stop...
```

```
                73            put   B220EXEC1
               >1    * OPcode execute phase dispatch table
               >2
               >3    optabl   equ   *          ; Low byte of execute routines
09EC: A0       >4             db    <HLT       ; S ---- 00 ---- HaLT
09ED: A0       >5             db    <NOP       ; S ---- 01 ---- No OP
09EE: C3       >6             db    <OPerr     ;        02
09EF: A3       >7             db    <PRD       ; S unnv 03 ADDR Pap tape RD
09F0: A9       >8             db    <PRB       ; S u--v 04 ADDR Pap tape Rd, Br
09F1: 35       >9             db    <PRI       ; S unnv 05 ADDR Pap tape Rd, Inv
09F2: 38       >10            db    <PWR       ; S unn- 06 ADDR Pap tape WR
09F3: 6E       >11            db    <PWI       ; S u--- 07 ADDR Pap tape Wr, Int
09F4: 5C       >12            db    <KAD       ; S ---- 08 ---- Keyboard ADd
09F5: 71       >13            db    <SPO       ; S dnnv 09 ADDR Sup Print Out
09F6: C3 C3 C3 >14            db    <OPerr,<OPerr,<OPerr,<OPerr,<OPerr,<OPerr
09FC: 0A       >15            db    <CAD       ; S ---v 10 ADDR Clear ADd (Abs)
09FD: F5       >16            db    <CSU       ; S ---v 11 ADDR Clear SUb (Abs)
09FE: 2A       >17            db    <ADD       ; S ---v 12 ADDR ADD (Abs)
09FF: BC       >18            db    <SUB       ; S ---v 13 ADDR SUBtract (Abs)
0A00: D2       >19            db    <MUL       ; S ---- 14 ADDR MULtiply
0A01: 5B       >20            db    <DIV       ; S ---- 15 ADDR DIVide
0A02: D6       >21            db    <RND       ; S ---- 16 ---- RouND
0A03: F8       >22            db    <EXT       ; S ---- 17 ADDR EXTract
0A04: 20       >23            db    <CFA       ; S sLfv 18 ADDR Comp Fld A (R)
0A05: 9A       >24            db    <ADL       ; S ---- 19 ADDR ADd to Location
0A06: C3 C3 C3 >25            db    <OPerr,<OPerr,<OPerr,<OPerr,<OPerr,<OPerr
0A0C: 86       >26            db    <IBB       ; S nnnn 20 ADDR Increase B, Br
0A0D: 99       >27            db    <DBB       ; S nnnn 21 ADDR Decrease B, Br
0A0E: DE       >28            db    <FAD       ; S n--v 22 ADDR Float ADd (Abs)
0A0F: EB       >29            db    <FSU       ; S n--v 23 ADDR Float SUb (Abs)
0A10: 00       >30            db    <FMU       ; S ---- 24 ADDR Float MUltiply
0A11: 9B       >31            db    <FDV       ; S ---- 25 ADDR Float DiVide
0A12: 1E       >32            db    <IFL       ; S sLnn 26 ADDR Inc Fld Loc
0A13: 64       >33            db    <DFL       ; S sLnn 27 ADDR Dec Fld Loc
0A14: 74       >34            db    <DLB       ; S sLnn 28 ADDR Dec fld loc,Ld B
0A15: 20       >35            db    <RTF       ; S -nn- 29 ADDR Record TransFer
0A16: C3 C3 C3 >36            db    <OPerr,<OPerr,<OPerr,<OPerr,<OPerr,<OPerr
0A1C: EF       >37            db    <BUN       ; S ---- 30 ADDR Branch UNcond
0A1D: AC       >38            db    <BOF       ; S ---- 31 ADDR Branch OverFlow
0A1E: B9       >39            db    <BRP       ; S ---- 32 ADDR Branch RePeat
0A1F: BF       >40            db    <BSA       ; S ---n 33 ADDR Branch Sign A
0A20: C9       >41            db    <BCH       ; S ---v 34 ADDR Br Comp Hi (Lo)
0A21: DD       >42            db    <BCE       ; S ---v 35 ADDR Br Comp Eq (Un)
0A22: 06       >43            db    <BFA       ; S sLnn 36 ADDR Branch Field A
0A23: 02       >44            db    <BFR       ; S sLnn 37 ADDR Branch Field R
0A24: 55       >45            db    <BCS       ; S u--- 38 ADDR Br Control Sw
0A25: 62       >46            db    <SOR       ; S ---V 39 ---- Set Ov Remember
0A26: C3 C3 C3 >47            db    <OPerr,<OPerr,<OPerr,<OPerr,<OPerr,<OPerr
0A2C: 76       >48            db    <STA       ; S sLfv 40 ADDR STore A (R/B)
0A2D: DD       >49            db    <LDR       ; S ---- 41 ADDR LoaD R
0A2E: E9       >50            db    <LDB       ; S ---v 42 ADDR LoaD B (Comp)
0A2F: 0F       >51            db    <LSA       ; S ---n 43 ---- Load Sign A
0A30: 18       >52            db    <STP       ; S ---- 44 ADDR STore P
0A31: 2D       >53            db    <CLA       ; S ---v 45 ---- CLr A/R/AR/B/AB/T
0A32: 4E       >54            db    <CLL       ; S ---- 46 ADDR CLear Location
0A33: C3       >55            db    <OPerr     ;        47
0A34: 59       >56            db    <SRA       ; S ---v 48 --nn Shft Rt A (AR/AS)
0A35: 8E       >57            db    <SLA       ; S ---v 49 --nn Shft Lt A (AR/AS)
0A36: C3 C3 C3 >58            db    <OPerr,<OPerr,<OPerr,<OPerr,<OPerr,<OPerr
0A3C: B0       >59            db    <MTS       ; S uhhv 50 addr Mag Tape Search
0A3D: 30       >60            db    <MTC       ; S uhhK 51 addr Mag Tape sCan
0A3E: AD       >61            db    <MRD       ; S un-v 52 addr Mag tape ReaD
0A3F: AC       >62            db    <MRR       ; S un-v 53 addr Mt Read Record
0A40: 9D       >63            db    <MIW       ; S unkk 54 addr Mt Init Write
```

```
                              ===== Page 46 =====

0A41: 9C       >64            db    <MIR       ; S un-- 55 addr Mt Init wr Rec
0A42: 1B       >65            db    <MOW       ; S unkk 56 addr Mt OverWrite
0A43: 1A       >66            db    <MOR       ; S un-- 57 addr Mt Overwr Rec
0A44: 76       >67            db    <MPF       ; S un-v 58 ---- Mt Pos Fwd
0A45: B3       >68            db    <MIB       ; S u--v 59 addr Mt Interr Branch
```

```
               >70   noAD     equ   $8000      ; Hi bit means "ignore ADDR"
               >71   operr    equ   OPerr+noAD ; Ignore ADDR on illegal OPs.
               >72
               >73   optabh   equ   *          ; High byte of execute routines
0A46: 8A       >74            db    >HLT+noAD  ; S ---- 00 ---- HaLT
0A47: 8A       >75            db    >NOP+noAD  ; S ---- 01 ---- No OP
0A48: 89       >76            db    >operr     ;        02
0A49: 0A       >77            db    >PRD       ; S unnv 03 ADDR Pap tape RD
0A4A: 0A       >78            db    >PRB       ; S u--v 04 ADDR Pap tape Rd, Br
0A4B: 0B       >79            db    >PRI       ; S unnv 05 ADDR Pap tape Rd, Inv
0A4C: 0B       >80            db    >PWR       ; S unn- 06 ADDR Pap tape WR
0A4D: 0B       >81            db    >PWI       ; S u--- 07 ADDR Pap tape Wr, Int
0A4E: 89       >82            db    >KAD+noAD  ; S ---- 08 ---- Keyboard ADd
0A4F: 0B       >83            db    >SPO       ; S dnnv 09 ADDR Sup Print Out
0A50: 89 89 89 >84            db    >operr,>operr,>operr,>operr,>operr,>operr
0A56: 0C       >85            db    >CAD       ; S ---v 10 ADDR Clear ADd (Abs)
0A57: 0B       >86            db    >CSU       ; S ---v 11 ADDR Clear SUbtr (Abs)
0A58: 0C       >87            db    >ADD       ; S ---v 12 ADDR ADD (Abs)
0A59: 0C       >88            db    >SUB       ; S ---v 13 ADDR SUBtract (Abs)
0A5A: 0C       >89            db    >MUL       ; S ---- 14 ADDR MULtiply
0A5B: 0D       >90            db    >DIV       ; S ---- 15 ADDR DIVide
0A5C: 8D       >91            db    >RND+noAD  ; S ---- 16 ---- RouND
0A5D: 0D       >92            db    >EXT       ; S ---- 17 ADDR EXTract
0A5E: 0E       >93            db    >CFA       ; S sLfv 18 ADDR Comp Fld A (R)
0A5F: 0C       >94            db    >ADL       ; S ---- 19 ADDR ADd to Location
0A60: 89 89 89 >95            db    >operr,>operr,>operr,>operr,>operr,>operr
0A66: 12       >96            db    >IBB       ; S nnnn 20 ADDR Increase B, Br
0A67: 12       >97            db    >DBB       ; S nnnn 21 ADDR Decrease B, Br
0A68: 0E       >98            db    >FAD       ; S n--v 22 ADDR Float ADd (Abs)
0A69: 0F       >99            db    >FSU       ; S n--v 23 ADDR Float SUb (Abs)
0A6A: 10       >100           db    >FMU       ; S ---- 24 ADDR Float MUltiply
0A6B: 10       >101           db    >FDV       ; S ---- 25 ADDR Float DiVide
0A6C: 11       >102           db    >IFL       ; S sLnn 26 ADDR Inc Fld Loc
0A6D: 11       >103           db    >DFL       ; S sLnn 27 ADDR Dec Fld Loc
0A6E: 11       >104           db    >DLB       ; S sLnn 28 ADDR Dec fld loc,Ld B
0A6F: 12       >105           db    >RTF       ; S -nn- 29 ADDR Record TransFer
0A70: 89 89 89 >106           db    >operr,>operr,>operr,>operr,>operr,>operr
0A76: 12       >107           db    >BUN       ; S ---- 30 ADDR Branch UNcond
0A77: 12       >108           db    >BOF       ; S ---- 31 ADDR Branch OverFlow
0A78: 12       >109           db    >BRP       ; S ---- 32 ADDR Branch RePeat
0A79: 12       >110           db    >BSA       ; S ---n 33 ADDR Branch Sign A
0A7A: 12       >111           db    >BCH       ; S ---v 34 ADDR Br Comp Hi (Lo)
0A7B: 12       >112           db    >BCE       ; S ---v 35 ADDR Br Comp Eq (Un)
0A7C: 13       >113           db    >BFA       ; S sLnn 36 ADDR Branch Field A
0A7D: 13       >114           db    >BFR       ; S sLnn 37 ADDR Branch Field R
0A7E: 13       >115           db    >BCS       ; S u--- 38 ADDR Br Control Sw
0A7F: 13       >116           db    >SOR       ; S ---v 39 ---- Set Ov Remember
0A80: 89 89 89 >117           db    >operr,>operr,>operr,>operr,>operr,>operr
0A86: 13       >118           db    >STA       ; S sLfv 40 ADDR STore A (R/B)
0A87: 13       >119           db    >LDR       ; S ---- 41 ADDR LoaD R
0A88: 13       >120           db    >LDB       ; S ---v 42 ADDR LoaD B (Comp)
0A89: 94       >121           db    >LSA+noAD  ; S ---n 43 ---- Load Sign A
0A8A: 14       >122           db    >STP       ; S ---- 44 ADDR STore P
0A8B: 94       >123           db    >CLA+noAD  ; S ---v 45 ---- CLr A/R/AR/B/AB/T
0A8C: 14       >124           db    >CLL       ; S ---- 46 ADDR CLear Location
0A8D: 89       >125           db    >operr     ;        47
0A8E: 94       >126           db    >SRA+noAD  ; S ---v 48 --nn Shft Rt A (AR/AS)
0A8F: 94       >127           db    >SLA+noAD  ; S ---v 49 --nn Shft Lt A (AR/AS)
0A90: 89 89 89 >128           db    >operr,>operr,>operr,>operr,>operr,>operr
0A96: 16       >129           db    >MTS       ; S uhhv 50 addr Mag Tape Search
0A97: 17       >130           db    >MTC       ; S uhhk 51 addr Mag Tape sCan
0A98: 17       >131           db    >MRD       ; S un-v 52 addr Mag tape ReaD
0A99: 17       >132           db    >MRR       ; S un-v 53 addr Mt Read Record
```

```
0A9A: 18        >133          db    >MIW       ; S unkk 54 addr Mt Init Write
0A9B: 18        >134          db    >MIR       ; S un-- 55 addr Mt Init wr Rec
0A9C: 19        >135          db    >MOW       ; S unkk 56 addr Mt OverWrite
0A9D: 19        >136          db    >MOR       ; S un-- 57 addr Mt Overwr Rec
0A9E: 99        >137          db    >MPF+noAD  ; S un-v 58 ---- Mt Pos Fwd
0A9F: 19        >138          db    >MIB       ; S u--v 59 addr Mt Interr Branch
```

```
                              ===== Page 49 =====

                 ***********************************************************
                 *                                                         *
                 *            B220 Instruction Execute Routines            *
                 *                                                         *
                 * For all OPs with ADDR = memory address, Y = 0           *
                 *  and A and rD+S = sign of MEM operand.                  *
                 *                                                         *
                 ***********************************************************

                 HLT      equ   *          ; Halt is executed in 'fetch'.

0AA0: 4C 01 09   NOP      jmp   fetch      ; Do nothing.

0AA3: 20 BD 0A   PRD      jsr   ]prd       ; Paper tape ReaD
0AA6: 4C 01 09            jmp   fetch

0AA9: A5 99      PRB      lda   rC+sL      ; Paper tape Read & Branch
0AAB: 29 F0               and   #$F0       ; Fake NN = 00 (100 words)
0AAD: 85 99               sta   rC+sL
0AAF: A5 9A               lda   rC+VV
0AB1: 29 0F               and   #$0F
0AB3: 09 01               ora   #$01       ;  and xeq sign 6/7.
0AB5: 85 9A               sta   rC+VV
0AB7: 20 BD 0A   :read    jsr   ]prd       ; Read "tape" until
0ABA: 4C B7 0A            jmp   :read      ;  sign 6/7 terminates.

                 Bmodflg  equ   linev      ; B-modification flag
                 xeqflg   equ   linev+1    ; Sign 6/7 execute flag

0ABD: 20 75 15   ]prd     jsr   midNN      ; Get word count (1..100)
0AC0: 85 D1               sta   NN         ;  in binary.
0AC2: A5 9A               lda   rC+VV      ; Examine variant digit
0AC4: 29 08               and   #$08       ; 8-bit on?
0AC6: 85 D4               sta   Bmodflg    ; Set B-modify mask.
0AC8: A5 9A               lda   rC+VV      ; Variant again...
0ACA: 29 01               and   #$01       ; Execute 6/7 sign?
0ACC: F0 02               beq   :noxeq     ; -No, ignore 6/7 sign.
0ACE: A9 06               lda   #6         ; -Yes, set xeq mask.
0AD0: 85 D5      :noxeq   sta   xeqflg
0AD2: A2 00               ldx   #PTRclass  ; PTRDR device class
0AD4: 20 58 08            jsr   M_iosel    ; Select device.
0AD7: 20 70 08   :readlp  jsr   M_getwrd   ; Next word to rD.
0ADA: A5 AA               lda   rD+S       ; Sign digit 8/9?
0ADC: 25 D4               and   Bmodflg    ; Variant 8-bit
0ADE: F0 05               beq   :noBmod    ; -No B modification.
0AE0: 20 12 0B            jsr   BmodrD     ; -B-modify address
0AE3: 10 08               bpl   :store     ; (always)

0AE5: A5 AA      :noBmod  lda   rD+S       ; Re-fetch sign digit
0AE7: 25 D5               and   xeqflg     ; Apply xeq mask (0/6)
0AE9: C9 06               cmp   #6         ; Sign = 6 or 7?
0AEB: F0 0B               beq   :xeq       ; -Yes, execute it.
0AED: 20 28 0B   :store   jsr   storerD    ; -No, store rD & adv memptr.
0AF0: C6 D1               dec   NN         ; More words?
0AF2: D0 E3               bne   :readlp    ; -Yes, continue scan.
0AF4: 20 64 08            jsr   M_iodsel   ; Deselect device.
0AF7: 60                  rts              ; -No, return.

0AF8: C5 AA      :xeq     cmp   rD+S       ; Is sign 6, or is it 7?
0AFA: F0 03               beq   :notB      ; =6, no B modification.
0AFC: 20 12 0B            jsr   BmodrD     ; =7, B modify.
0AFF: A2 05      :notB    ldx   #5         ; Execute input word.
0B01: B5 AA      :xeqlp   lda   rD,x       ; Copy rD to rC.
0B03: 95 98               sta   rC,x
0B05: CA                  dex
0B06: 10 F9               bpl   :xeqlp
0B08: 86 C6               stx   skipincP   ; Don't inc P reg.
```

```
0B0A: 20 64 08 >207           jsr   M_iodsel   ; Deselect device.
0B0D: 68       >208           pla              ; No return.
0B0E: 68       >209           pla
0B0F: 4C 22 09 >210           jmp   execute    ; Execute instruction.
               >211
0B12: F8       >212  BmodrD   sed              ; / Decimal mode.
0B13: 18       >213           clc
0B14: A5 AF    >214           lda   rD+ADDR+1  ; Add rB to rD ADDR.
0B16: 65 95    >215           adc   rB+1
0B18: 85 AF    >216           sta   rD+ADDR+1
0B1A: A5 AE    >217           lda   rD+ADDR
0B1C: 65 94    >218           adc   rB
0B1E: 85 AE    >219           sta   rD+ADDR
0B20: D8       >220           cld              ; \ Binary mode.
0B21: A5 AA    >221           lda   rD+S       ; Turn off
0B23: 29 01    >222           and   #$01       ;  8-bit of sign.
0B25: 85 AA    >223           sta   rD+S       ; (return w/ >=)
0B27: 60       >224           rts
               >225
0B28: A0 05    >226  storerD  ldy   #5         ; Store rD
0B2A: B9 AA 00 >227  :stlp    lda   rD,y
0B2D: 91 CA    >228           sta   (memptr),y
0B2F: 88       >229           dey
0B30: 10 F8    >230           bpl   :stlp
0B32: 4C CB 08 >231           jmp   incmem     ; Inc memptr and return.
               >232
0B35: 4C C3 09 >233  PRI      jmp   OPerr      ; Unimplemented
```

===== Page 51 =====

```
0B38: 20 75 15 >235  PWR      jsr   midNN      ; Get word count
0B3B: 85 D1    >236           sta   NN         ;  in binary.
0B3D: A2 02    >237           ldx   #PTPclass  ; PTPCH device class.
0B3F: 20 58 08 >238           jsr   M_iosel    ; Select device.
0B42: 20 61 0B >239  :wrdlp   jsr   loadrD     ; (memptr) word --> rD
0B45: 20 7C 08 >240           jsr   M_putwrd   ; Put rD in buffer.
0B48: C6 D1    >241           dec   NN         ; More words?
0B4A: D0 F6    >242           bne   :wrdlp     ; -Yes, go again.
0B4C: 20 64 08 >243           jsr   M_iodsel   ; -No, deselect device.
0B4F: A9 EF    >244           lda   #EOF       ; Set End-Of-File flag.
0B51: A0 00    >245           ldy   #0
0B53: 8D 04 C0 >246           sta   WRITMAIN
0B56: 91 CC    >247           sta   (ptr),y
0B58: 8D 05 C0 >248           sta   WRITAUX
0B5B: 4C 01 09 >249           jmp   fetch
               >250
0B5E: 4C D3 09 >251  :ioerr   jmp   IOerr      ; Relay jump.
               >252
0B61: A0 05    >253  loadrD   ldy   #5         ; Load (memptr) into rD.
0B63: B1 CA    >254  :ldlp    lda   (memptr),y
0B65: 99 AA 00 >255           sta   rD,y
0B68: 88       >256           dey
0B69: 10 F8    >257           bpl   :ldlp
0B6B: 4C CB 08 >258           jmp   incmem     ; Adv to next word & return.
               >259
0B6E: 4C C3 09 >260  PWI      jmp   OPerr      ; Unimplemented
               >261
               >262  KAD      equ   ]stop      ; Kluge to allow rA mod.
```

```
0B71: 20 75 15 >264  SPO      jsr   midNN      ; Get count (NN) in A
0B74: 85 D1    >265           sta   NN         ; NN = binary word count.
0B76: A0 00    >266  :nxword  ldy   #0
0B78: B1 CA    >267           lda   (memptr),y ; Get sign
0B7A: C9 02    >268           cmp   #2         ; Alphanumeric?
0B7C: D0 3A    >269           bne   :num       ; -No, numeric.
0B7E: C8       >270  :nxchar  iny              ; -Yes, print alpha.
0B7F: B1 CA    >271           lda   (memptr),y ; Get next char
0B81: C9 26    >272           cmp   #$26       ; "Tab" code?
0B83: F0 11    >273           beq   :tab       ; -Yes, do tab.
0B85: C9 02    >274           cmp   #$02       ; -No, "Ignore" code?
0B87: F0 07    >275           beq   :ignore    ; -Yes, skip it.
0B89: AA       >276           tax              ; -No, translate B220
0B8A: BD 26 16 >277           lda   b220asc,x  ;       char to ASCII.
0B8D: 20 B8 08 >278           jsr   M_COUT     ;        and print it.
0B90: C0 05    >279  :ignore  cpy   #5         ; Word complete?
0B92: D0 EA    >280           bne   :nxchar    ; -No, keep going.
0B94: F0 4E    >281           beq   :done      ; -Yes, word done (always)
               >282
0B96: A2 00    >283  :tab     ldx   #0
0B98: A5 24    >284           lda   CH
0B9A: DD F0 0B >285  :nxtab   cmp   tabs,x     ; Find first tab
0B9D: 90 07    >286           bcc   :gottab    ;  greater than CH.
0B9F: E8       >287           inx
0BA0: E0 05    >288           cpx   #5
0BA2: D0 F6    >289           bne   :nxtab
0BA4: F0 EA    >290           beq   :ignore    ; (always) Skip if past tabs.
               >291
0BA6: 84 D0    >292  :gottab  sty   t1         ; Save Y
0BA8: BC F0 0B >293           ldy   tabs,x     ; Get target tab position.
0BAB: A9 A0    >294  :prtblnk lda   #"         "
0BAD: 20 B8 08 >295           jsr   M_COUT     ; Print blanks until at
0BB0: C4 24    >296           cpy   CH         ;  target tab position.
0BB2: D0 F7    >297           bne   :prtblnk
0BB4: A4 D0    >298           ldy   t1         ; Restore Y
0BB6: D0 D8    >299           bne   :ignore    ;  and continue. (always)
               >300
0BB8: A2 A0    >301  :num     ldx   #"         " ; Print blank if sign 0
0BBA: C9 00    >302           cmp   #0
0BBC: F0 09    >303           beq   :prtsign
0BBE: A2 AD    >304           ldx   #"-"       ; Print - if sign 1
0BC0: C9 01    >305           cmp   #1
0BC2: F0 03    >306           beq   :prtsign
0BC4: 09 B0    >307           ora   #"0"       ; Else print sign digit.
0BC6: AA       >308           tax
0BC7: 8A       >309  :prtsign txa
0BC8: 20 B8 08 >310           jsr   M_COUT
0BCB: C8       >311  :nxbyte  iny              ; Print rest of number.
0BCC: B1 CA    >312           lda   (memptr),y
0BCE: 48       >313           pha
0BCF: 4A       >314           lsr
0BD0: 4A       >315           lsr
0BD1: 4A       >316           lsr
0BD2: 4A       >317           lsr              ; Hi digit in A
0BD3: 09 B0    >318           ora   #"0"       ; OR in zone
0BD5: 20 B8 08 >319           jsr   M_COUT     ;  and print digit.
0BD8: 68       >320           pla              ; Recover low digit
0BD9: 29 0F    >321           and   #$0F       ; Isolate it
0BDB: 09 B0    >322           ora   #"0"       ;  add zone
0BDD: 20 B8 08 >323           jsr   M_COUT     ;   and print it.
0BE0: C0 05    >324           cpy   #5         ; End of word?
0BE2: D0 E7    >325           bne   :nxbyte    ; -No, continue.
0BE4: C6 D1    >326  :done    dec   NN         ; -Yes, more words?
0BE6: F0 05    >327           beq   :quit      ; -No, all done.
0BE8: 20 CB 08 >328           jsr   incmem     ; -Yes, increment memptr.
0BEB: D0 89    >329           bne   :nxword    ; (always)
               >330
```

```
0BED: 4C 01 09  :quit    jmp   fetch

0BF0: 09 11 19  tabs     db    9,17,25,33,41 ; SPO tab table

0BF5: A5 9A     CSU      lda   rC+VV       ; CSU/CSA
0BF7: 29 0F              and   #$0F        ; Isolate variant digit.
0BF9: C9 01              cmp   #$01        ; CSA?
0BFB: D0 06              bne   :csu        ; -No, CSU.
0BFD: A5 AA              lda   rD+S        ; -Yes, CSA.
0BFF: 09 01              ora   #$01        ;   Force sign negative.
0C01: D0 11              bne   loadrA      ; (always)

0C03: A5 AA     :csu     lda   rD+S        ; CSU
0C05: 49 01              eor   #$01        ; Flip the 1-bit
0C07: 4C 14 0C           jmp   loadrA      ;  and complete the load.


0C0A: A5 9A     CAD      lda   rC+VV       ; CAD/CAA
0C0C: 29 0F              and   #$0F        ; Isolate variant digit.
0C0E: C9 01              cmp   #$01        ; CAA?
0C10: F0 11              beq   CAA         ; -Yes.
0C12: A5 AA              lda   rD+S        ; -No, CAD.  Sign unchanged.
0C14: 85 9E     loadrA   sta   rA+S        ; Set rA sign.
0C16: A0 05              ldy   #5
0C18: B1 CA     :cpyloop lda   (memptr),y
0C1A: 99 9E 00           sta   rA,y
0C1D: 88                 dey
0C1E: D0 F8              bne   :cpyloop
0C20: 4C 01 09           jmp   fetch

0C23: A5 AA     CAA      lda   rD+S        ; CAA
0C25: 29 FE              and   #$FE        ; Force sign positive
0C27: 4C 14 0C           jmp   loadrA      ;  and complete the load.
```

```
0C2A: A5 9A    >365  ADD      lda   rC+VV      ; ADD, ADA
0C2C: 29 0F    >366           and   #$0F
0C2E: C9 01    >367           cmp   #1         ; ADA?
0C30: D0 04    >368           bne   :add       ; -No, ADD.
0C32: A9 00    >369           lda   #0         ; -Yes, force MEM sign +
0C34: 85 AA    >370           sta   rD+S
0C36: 20 3C 0C >371  :add     jsr   ]add       ; Do the add.
0C39: 4C 01 09 >372           jmp   fetch
               >373
0C3C: A5 9E    >374  ]add     lda   rA+S
0C3E: 29 01    >375           and   #$01
0C40: 85 9E    >376           sta   rA+S       ; Force sign 0 (+) or 1 (-)
0C42: 45 AA    >377           eor   rD+S       ; Signs same or different?
0C44: 29 01    >378           and   #$01
0C46: D0 18    >379           bne   :subtr     ; -Different, subtract.
0C48: A0 05    >380           ldy   #5         ; -Same, add.
0C4A: F8       >381           sed              ; / Decimal mode.
0C4B: 18       >382           clc
0C4C: B9 9E 00 >383  :addloop lda   rA,y       ; Do the addition...
0C4F: 71 CA    >384           adc   (memptr),y
0C51: 99 9E 00 >385           sta   rA,y
0C54: 88       >386           dey
0C55: D0 F5    >387           bne   :addloop
0C57: D8       >388           cld              ; \ Back to binary.
0C58: 90 3F    >389           bcc   :done      ; Done.
               >390           seti  Ov         ; Signal Overflow
0C5A: A9 FF    >390           lda   #$FF
0C5C: 85 C3    >390           sta   Ov         ; Set non-zero.
               >390           eom
0C5E: D0 39    >391           bne   :done      ; (always)
               >392
0C60: A0 01    >393  :subtr   ldy   #1         ; Compare magnitudes.
0C62: B9 9E 00 >394  :comloop lda   rA,y
0C65: D1 CA    >395           cmp   (memptr),y
0C67: F0 04    >396           beq   :cont      ; Equal, keep comparing.
0C69: B0 07    >397           bcs   :Abig      ; rA is bigger
0C6B: 90 16    >398           bcc   :Asmall    ; rA is smaller
               >399
0C6D: C8       >400  :cont    iny
0C6E: C0 06    >401           cpy   #6
0C70: D0 F0    >402           bne   :comloop   ; If =, fall into :Abig.
0C72: A0 05    >403  :Abig    ldy   #5         ; Subtract MEM from rA.
0C74: F8       >404           sed              ; / Decimal mode.
0C75: B9 9E 00 >405  :subloop lda   rA,y
0C78: F1 CA    >406           sbc   (memptr),y
0C7A: 99 9E 00 >407           sta   rA,y
0C7D: 88       >408           dey
0C7E: D0 F5    >409           bne   :subloop
0C80: D8       >410           cld              ; \ Back to binary.
0C81: F0 16    >411           beq   :done      ; (always)
               >412
0C83: A5 AA    >413  :Asmall  lda   rD+S       ; MEM - rA ==> rA
0C85: 29 01    >414           and   #$01       ; rA sign = MEM sign.
0C87: 85 9E    >415           sta   rA+S
0C89: A0 05    >416           ldy   #5
0C8B: F8       >417           sed              ; / Decimal mode.
0C8C: 38       >418           sec
0C8D: B1 CA    >419  :sloop   lda   (memptr),y
0C8F: F9 9E 00 >420           sbc   rA,y
0C92: 99 9E 00 >421           sta   rA,y
0C95: 88       >422           dey
0C96: D0 F5    >423           bne   :sloop
0C98: D8       >424           cld              ; \ Back to binary.
0C99: 60       >425  :done    rts
```

```
0C9A: A5 9E    >427  ADL      lda   rA+S        ; Force rA sign
0C9C: 29 01    >428           and   #$01        ;  to 0 or 1.
0C9E: 85 9E    >429           sta   rA+S
0CA0: A2 FA    >430           ldx   #-6         ; MEM + rA ==> MEM
0CA2: B5 A4    >431  :pushlp  lda   rA+6,x      ; Push rA
0CA4: 48       >432           pha
0CA5: E8       >433           inx
0CA6: D0 FA    >434           bne   :pushlp
0CA8: 20 3C 0C >435           jsr   ]add        ; rA + MEM ==> rA
0CAB: A0 05    >436           ldy   #5          ; rA ==> MEM
0CAD: B9 9E 00 >437  :mvloop  lda   rA,y
0CB0: 91 CA    >438           sta   (memptr),y
0CB2: 68       >439           pla               ;  and pop rA.
0CB3: 99 9E 00 >440           sta   rA,y
0CB6: 88       >441           dey
0CB7: 10 F4    >442           bpl   :mvloop
0CB9: 4C 01 09 >443           jmp   fetch
               >444
0CBC: A5 9A    >445  SUB      lda   rC+VV       ; SUB, SUA
0CBE: 29 0F    >446           and   #$0F
0CC0: C9 01    >447           cmp   #1          ; SUA?
0CC2: F0 06    >448           beq   :setsign    ; -Yes, force operand neg.
0CC4: A5 AA    >449  :sub     lda   rD+S        ; -No, SUB.
0CC6: 29 01    >450           and   #$01        ; Invert
0CC8: 49 01    >451           eor   #$01        ;  operand
0CCA: 85 AA    >452  :setsign sta   rD+S        ;   sign
0CCC: 20 3C 0C >453           jsr   ]add        ;    and add.
0CCF: 4C 01 09 >454           jmp   fetch
```

```
0CD2: 20 D8 0C >456  MUL      jsr   multiply   ; Multiply
0CD5: 4C 01 09 >457           jmp   fetch
               >458
0CD8: 45 9E    >459  multiply eor   rA+S       ; Multiply subroutine
0CDA: 29 01    >460           and   #$01
0CDC: 48       >461           pha              ; Save result sign
0CDD: A2 00    >462           ldx   #0
0CDF: A0 05    >463           ldy   #5
0CE1: B1 CA    >464  :init    lda   (memptr),y ; rD = multiplicand
0CE3: 99 AA 00 >465           sta   rD,y
0CE6: 99 B0 00 >466           sta   rD10,y     ; rD10 = multiplicand
0CE9: B9 9E 00 >467           lda   rA,y       ; rR = multiplier
0CEC: 99 A4 00 >468           sta   rR,y
0CEF: 96 9E    >469           stx   rA,y       ; rA = 0 (including sign)
0CF1: 88       >470           dey
0CF2: 10 ED    >471           bpl   :init
0CF4: A5 C3    >472           lda   Ov         ; FMU overflow pending?
0CF6: C9 80    >473           cmp   #$80
0CF8: D0 02    >474           bne   :cont      ; -No, continue.
0CFA: 68       >475           pla              ; -Yes, discard result sign
0CFB: 60       >476           rts              ;   and return.
               >477
0CFC: 86 AA    >478  :cont    stx   rD+S       ; Clear rD sign
0CFE: 86 B0    >479           stx   rD10+S     ;  and rD10 sign.
0D00: A0 04    >480           ldy   #4         ; 4 bits/digit.
0D02: 18       >481  :shloop  clc              ; Shift in zeros.
0D03: 26 B5    >482           rol   rD10+5     ; Multiply rD10 by 10.
0D05: 26 B4    >483           rol   rD10+4
0D07: 26 B3    >484           rol   rD10+3
0D09: 26 B2    >485           rol   rD10+2
0D0B: 26 B1    >486           rol   rD10+1
0D0D: 26 B0    >487           rol   rD10
0D0F: 88       >488           dey
0D10: D0 F0    >489           bne   :shloop
0D12: A9 05    >490           lda   #5         ; Set multiplier byte
0D14: 85 D0    >491           sta   t1         ;  count = 5.
0D16: F8       >492           sed              ; / Decimal mode.
0D17: A5 A9    >493  :ckadd1  lda   rR+5
0D19: 29 0F    >494           and   #$0F       ; Low digit of multiplier
0D1B: F0 10    >495           beq   :ckadd10   ; Skip add1 if zero.
0D1D: A8       >496           tay              ; Y = add1 count.
0D1E: A2 05    >497  :add1    ldx   #5
0D20: 18       >498           clc              ; rA = rA + rD
0D21: B5 9E    >499  :add1lp  lda   rA,x
0D23: 75 AA    >500           adc   rD,x
0D25: 95 9E    >501           sta   rA,x
0D27: CA       >502           dex
0D28: 10 F7    >503           bpl   :add1lp
0D2A: 88       >504           dey              ; More adds?
0D2B: D0 F1    >505           bne   :add1      ; -Yes.
0D2D: A5 A9    >506  :ckadd10 lda   rR+5       ; Low multiplier byte
0D2F: 29 F0    >507           and   #$F0       ; High digit of byte
0D31: F0 14    >508           beq   :shift     ; Skip add10 if zero.
0D33: 4A       >509           lsr
0D34: 4A       >510           lsr
0D35: 4A       >511           lsr
0D36: 4A       >512           lsr
0D37: A8       >513           tay              ; Y = add10 count.
0D38: A2 05    >514  :add10   ldx   #5
0D3A: 18       >515           clc              ; rA = rA + rD10
0D3B: B5 9E    >516  :add10lp lda   rA,x
0D3D: 75 B0    >517           adc   rD10,x
0D3F: 95 9E    >518           sta   rA,x
0D41: CA       >519           dex
0D42: 10 F7    >520           bpl   :add10lp
0D44: 88       >521           dey              ; More adds?
0D45: D0 F1    >522           bne   :add10     ; -Yes.
```

```
0D47: 20 1D 15 >523  :shift   jsr   srT2       ; -No, shift |rA| & |rR|
0D4A: A5 9E    >524           lda   rA+S       ;  right 2 digits
0D4C: 85 9F    >525           sta   rA+1       ;   including rA sign.
0D4E: 86 9E    >526           stx   rA+S       ; Clear rA sign.
0D50: C6 D0    >527           dec   t1         ; Keep going if more
0D52: D0 C3    >528           bne   :ckadd1    ;  multiplier digits.
0D54: D8       >529           cld              ; \ Back to binary.
0D55: 68       >530           pla              ; Recover product sign
0D56: 85 9E    >531           sta   rA+S       ;  and set rA & rR signs.
0D58: 85 A4    >532           sta   rR+S
0D5A: 60       >533           rts
```

```
0D5B: 20 61 0D >535  DIV      jsr   divide     ; DIVide
0D5E: 4C 01 09 >536           jmp   fetch
               >537
0D61: 45 9E    >538  divide   eor   rA+S
0D63: 29 01    >539           and   #$01
0D65: 48       >540           pha              ; Sign of quotient
0D66: A5 9E    >541           lda   rA+S
0D68: 85 A4    >542           sta   rR+S       ; Sign of remainder
0D6A: C8       >543           iny              ; Y = 1: skip signs.
0D6B: B9 9E 00 >544  :comp    lda   rA,y       ; Compare rA magnitude
0D6E: D1 CA    >545           cmp   (memptr),y ;  with divisor magnitude.
0D70: 90 0D    >546           bcc   :divide    ; rA < MEM, so divide.
0D72: D0 05    >547           bne   :oflow     ; rA > MEM, overflow.
0D74: C8       >548           iny
0D75: C0 06    >549           cpy   #6
0D77: D0 F2    >550           bne   :comp
               >551  :oflow   seti  Ov         ; Signal overflow
0D79: A9 FF    >551           lda   #$FF
0D7B: 85 C3    >551           sta   Ov         ; Set non-zero.
               >551           eom
0D7D: 68       >552           pla              ; Drop result sign
0D7E: 60       >553           rts              ;  and return.
               >554
0D7F: A0 0A    >555  :divide  ldy   #10        ; Quotient digit count = 10.
0D81: 84 D0    >556           sty   t1
0D83: A0 05    >557           ldy   #5
0D85: B1 CA    >558  :div2rD  lda   (memptr),y ; Move divisor to rD
0D87: 99 AA 00 >559           sta   rD,y
0D8A: 88       >560           dey
0D8B: D0 F8    >561           bne   :div2rD
0D8D: 84 9E    >562           sty   rA+S       ; Clear sign of rA
0D8F: 84 AA    >563           sty   rD+S       ;  and rD.
0D91: F8       >564           sed              ; / Decimal mode.
0D92: A0 04    >565  :shift   ldy   #4         ; 4 bits/digit.
0D94: 18       >566  :shiftlp clc              ; Shift AR left 1 digit
0D95: 20 31 15 >567           jsr   slT        ;  shifting in zeros.
0D98: 26 9E    >568           rol   rA+S       ;   (include sign in A)
0D9A: 88       >569           dey
0D9B: D0 F7    >570           bne   :shiftlp
0D9D: A2 00    >571           ldx   #0
0D9F: B5 9E    >572  :complp  lda   rA,x       ; Compare A with divisor
0DA1: D5 AA    >573           cmp   rD,x
0DA3: 90 25    >574           bcc   :zero      ; Speed up quotient zeros.
0DA5: D0 05    >575           bne   :sub       ; A > divisor
0DA7: E8       >576           inx
0DA8: E0 06    >577           cpx   #6
0DAA: D0 F3    >578           bne   :complp
0DAC: A2 05    >579  :sub     ldx   #5         ; A(ext) = A(ext) - D(ext).
0DAE: 38       >580           sec
0DAF: B5 9E    >581  :sublp   lda   rA,x
0DB1: F5 AA    >582           sbc   rD,x
0DB3: 95 9E    >583           sta   rA,x
0DB5: CA       >584           dex
0DB6: 10 F7    >585           bpl   :sublp
0DB8: 90 04    >586           bcc   :restore   ; Restore if underflow
0DBA: E6 A9    >587           inc   rR+5       ; Increment quotient digit.
0DBC: D0 EE    >588           bne   :sub       ; (always)
               >589
0DBE: A2 05    >590  :restore ldx   #5         ; Add divisor back to A.
0DC0: 18       >591           clc
0DC1: B5 9E    >592  :restlp  lda   rA,x
0DC3: 75 AA    >593           adc   rD,x
0DC5: 95 9E    >594           sta   rA,x
0DC7: CA       >595           dex
0DC8: 10 F7    >596           bpl   :restlp
0DCA: C6 D0    >597  :zero    dec   t1         ; Quotient complete?
0DCC: D0 C4    >598           bne   :shift     ; -No, keep dividing.
```

```
0DCE: 20 46 15 >599           jsr   exchAR     ; -Yes, exchange A and R
0DD1: D8       >600           cld              ; \ Back to binary.
0DD2: 68       >601           pla
0DD3: 85 9E    >602           sta   rA+S       ; Set quotient sign.
0DD5: 60       >603           rts
```

```
0DD6: A5 A5    >605  RND      lda   rR+1       ; Hi digit of rR
0DD8: C9 50    >606           cmp   #$50       ; C=1 if hi digit >= 5.
0DDA: A2 A4    >607           ldx   #rR        ; Clear rR.
0DDC: 20 68 15 >608           jsr   clear      ; (Doesn't disturb C)
0DDF: 90 14    >609           bcc   :done      ; Done if hi digit < 5.
0DE1: F8       >610           sed              ; / Decimal mode.
0DE2: 38       >611           sec              ; Add 1 to rA.
0DE3: A2 05    >612           ldx   #5
0DE5: B5 9E    >613  :rndloop lda   rA,x
0DE7: 69 00    >614           adc   #0
0DE9: 95 9E    >615           sta   rA,x
0DEB: CA       >616           dex
0DEC: D0 F7    >617           bne   :rndloop
0DEE: D8       >618           cld              ; \ Back to binary.
0DEF: 90 04    >619           bcc   :done
               >620           seti  Ov         ; Signal Overflow.
0DF1: A9 FF    >620           lda   #$FF
0DF3: 85 C3    >620           sta   Ov         ; Set non-zero.
               >620           eom
0DF5: 4C 01 09 >621  :done    jmp   fetch
               >622
0DF8: A0 05    >623  EXT      ldy   #5         ; Extract digits from rA
0DFA: B1 CA    >624  :extlp   lda   (memptr),y ;  where MEM digits are odd.
0DFC: 29 11    >625           and   #$11       ; Isolate odd bits
0DFE: AA       >626           tax              ; $00, $01, $10, $11.
0DFF: BD 0E 0E >627           lda   :exttbl,x  ; $00, $0F, $F0, $FF.
0E02: 39 9E 00 >628           and   rA,y       ; Mask rA digits
0E05: 99 9E 00 >629           sta   rA,y
0E08: 88       >630           dey
0E09: 10 EF    >631           bpl   :extlp
0E0B: 4C 01 09 >632           jmp   fetch
               >633
0E0E: 00 0F    >634  :exttbl  db    $00,$0F    ; Indices $00, $01 used
0E10: 03 02 01 >635  signtbl  db    3,2,1,0,7,6,5,4,8,9 ; CFx sign order
0E1A: 00 00 00 >636           db    0,0,0,0    ; (filler)
0E1E: F0 FF    >637           db    $F0,$FF    ; Indices $10, $11 used.
               >638
0E20: A5 9A    >639  CFA      lda   rC+VV      ; CFA, CFR
0E22: A2 A4    >640           ldx   #rR
0E24: 29 01    >641           and   #$01       ; CFR?
0E26: D0 02    >642           bne   :cfr       ; -Yes.
0E28: A2 9E    >643           ldx   #rA        ; No, CFA.
0E2A: A5 9A    >644  :cfr     lda   rC+VV      ; Reload variant
0E2C: 29 10    >645           and   #$10       ; Partial field bit
0E2E: A8       >646           tay              ;  to Y.
0E2F: A9 D0    >647           lda   #BNEop     ; Do signed compare.
0E31: 20 40 0E >648           jsr   compare
0E34: 85 C2    >649           sta   COMP       ; Set COMPare indicator
0E36: A5 C1    >650           lda   ERR        ; Error detected?
0E38: D0 03    >651           bne   :err       ; -Yes, report it.
0E3A: 4C 01 09 >652           jmp   fetch
               >653
0E3D: 4C DB 09 >654  :err     jmp   ]err
```

```
                >656  **********************************************************
                >657  *                                                        *
                >658  * Compare register with (memptr), whole or partial field.*
                >659  *                                                        *
                >660  * Entry: X = Register addr, (memptr) = comparand addr    *
                >661  *        Y = Whole (0) or partial (not 0)                *
                >662  *        A = BNE (signed comp) or BCS (unsigned comp)    *
                >663  *                                                        *
                >664  *  Exit: A = COMP indicator state (<0, 0, >0)            *
                >665  *                                                        *
                >666  **********************************************************
                >667
0E40: 8D 6A 0E >668  compare  sta   :magonly   ; Signed/unsigned (BNE, BCS)
0E43: B5 00    >669           lda   0,x        ; Save register sign
0E45: 8D 6D 0E >670           sta   :cmpsign+1 ;  for compare.
0E48: 8E 9C 0E >671           stx   :comp1+1   ; And save register
0E4B: 8E C7 0E >672           stx   :comp2+1   ;  address for loads.
0E4E: 8E D2 0E >673           stx   :byte+1
0E51: 84 D1    >674           sty   NN         ; Save whole/partial.
0E53: C0 00    >675           cpy   #0         ; Whole/partial (0, not 0)
0E55: D0 06    >676           bne   :partial   ; -Yes.
0E57: A9 00    >677           lda   #0         ; -No, fake 0:0 field
0E59: A2 0B    >678           ldx   #11        ;   and compare signs.
0E5B: D0 0F    >679           bne   :cmpsign   ; (always)
               >680
0E5D: 20 54 15 >681  :partial jsr   splitsL    ; Split sL: A = s and X = L.
0E60: 18       >682           clc              ; A = low digit, 1..10
0E61: 69 01    >683           adc   #1         ;     low dig + 1, 2..11
0E63: 38       >684           sec
0E64: 86 D0    >685           stx   t1         ; Digit length
0E66: E5 D0    >686           sbc   t1         ; A = hi digit #
0E68: 90 18    >687           bcc   :flderr    ; <0 ==> Field error.
0E6A: D0 1F    >688  :magonly bne   :comp      ; >0 ==> Comp magnitudes.
0E6C: A0 00    >689  :cmpsign ldy   #0*0       ; =0 ==> Compare signs.
0E6E: C4 AA    >690           cpy   rD+S       ; Reg sign = MEM sign?
0E70: F0 15    >691           beq   :nosign    ; -Yes, comp magnitudes.
0E72: B9 10 0E >692           lda   signtbl,y  ; -No, translate reg sign
0E75: A4 AA    >693           ldy   rD+S       ; MEM sign
0E77: BE 10 0E >694           ldx   signtbl,y  ;  translated.
0E7A: 86 D0    >695           stx   t1
0E7C: C5 D0    >696           cmp   t1         ; Compare signs.
0E7E: E6 D1    >697           inc   NN         ; Force no flip.
0E80: D0 26    >698           bne   :neql      ; (always) Sign determines.
               >699
0E82: A5 C6    >700  :flderr  lda   "F"        ; Signal Field error.
0E84: 85 C1    >701           sta   ERR
0E86: 60       >702           rts
               >703
0E87: 18       >704  :nosign  clc              ; Exclude sign from field
0E88: 69 01    >705           adc   #1         ; Field start + 1
0E8A: CA       >706           dex              ; Field length - 1
0E8B: 18       >707  :comp    clc
0E8C: 69 01    >708           adc   #1
0E8E: 4A       >709           lsr              ; A = hi byte for compare
0E8F: A8       >710           tay              ; Y = hi byte index
0E90: B0 2E    >711           bcs   :lodigit   ; C ==> lo digit of hi byte.
0E92: CA       >712  :hidigit dex              ; Next digit, too?
0E93: D0 3C    >713           bne   :byte      ; -Yes, comp whole byte.
0E95: B1 CA    >714           lda   (memptr),y ; MEM byte
0E97: 29 F0    >715           and   #$F0       ; -No, final digit.
0E99: 85 D0    >716           sta   t1
0E9B: B9 00 00 >717  :comp1   lda   0*0,y      ; Reg byte
0E9E: 29 F0    >718           and   #$F0       ; Hi digit
0EA0: C5 D0    >719  :final   cmp   t1         ; Compare final digit.
0EA2: D0 04    >720  :done    bne   :neql      ; =?
0EA4: A9 00    >721           lda   #0         ; -Yes, A = 0.
0EA6: F0 06    >722           beq   :fin       ; (always)
```

```
                  >723
0EA8: A9 01       >724  :neql    lda   #1
0EAA: B0 02       >725           bcs   :fin        ; >
0EAC: A9 FF       >726           lda   #-1         ; <
0EAE: A4 D1       >727  :fin     ldy   NN          ; Recover whole/partial
0EB0: D0 0D       >728           bne   :noflip     ; Partial ==> no flip
0EB2: A6 AA       >729           ldx   rD+S        ; Original sign
0EB4: F0 09       >730           beq   :noflip     ; + if 0.
0EB6: E0 04       >731           cpx   #4          ; Collate as + or -?
0EB8: B0 05       >732           bcs   :noflip     ; + if >= 4.
0EBA: AA          >733           tax               ; - if 1, 2, or 3.
0EBB: F0 02       >734           beq   :noflip     ; Comp =, no flip.
0EBD: 49 80       >735           eor   #$80        ; Exchange > and <.
0EBF: 60          >736  :noflip  rts
                  >737
0EC0: B1 CA       >738  :lodigit lda   (memptr),y ; MEM byte
0EC2: 29 0F       >739           and   #$0F        ; Lo digit
0EC4: 85 D0       >740           sta   t1          ; Save for compare.
0EC6: B9 00 00 >741  :comp2   lda   0*0,y       ; Reg byte
0EC9: 29 0F       >742           and   #$0F        ; Lo digit
0ECB: C5 D0       >743           cmp   t1          ; Compare digits.
0ECD: D0 D3       >744           bne   :done       ; Done if unequal.
0ECF: F0 07       >745           beq   :nxbyte     ; Else continue (always)
                  >746
0ED1: B9 00 00 >747  :byte    lda   0*0,y       ; Reg byte
0ED4: D1 CA       >748           cmp   (memptr),y ; Compare w MEM.
0ED6: D0 CA       >749           bne   :done       ; Done if unequal.
0ED8: C8          >750  :nxbyte  iny               ; Advance byte index and
0ED9: CA          >751           dex               ;  decrement digit count
0EDA: D0 B6       >752           bne   :hidigit    ; Continue if digits left,
0EDC: F0 C4       >753           beq   :done       ;  else done. (always)
```

```
                74           put   B220EXEC2
0EDE: 29 01    >1    FAD     and   #$01        ; Standardize sign of
0EE0: 85 AA    >2            sta   rD+S        ;  MEM operand (0/1).
0EE2: A5 9A    >3            lda   rC+VV       ; FAD or FAA?
0EE4: 29 0F    >4            and   #$0F
0EE6: 49 01    >5            eor   #$01
0EE8: D0 02    >6            bne   ]fad        ; -FAD, continue.
0EEA: 85 AA    >7            sta   rD+S        ; -FAA, force +.
0EEC: A5 99    >8    ]fad    lda   rC+sL       ; Get normalization limit.
0EEE: 4A       >9            lsr
0EEF: 4A       >10           lsr
0EF0: 4A       >11           lsr
0EF1: 4A       >12           lsr
0EF2: D0 02    >13           bne   :nonzero
0EF4: A9 0A    >14           lda   #10
0EF6: 85 D1    >15   :nonzero sta  NN          ; Save binary norm limit.
0EF8: A5 9E    >16           lda   rA+S        ; Standardize rA sign (0/1)
0EFA: 29 01    >17           and   #$01
0EFC: 85 9E    >18           sta   rA+S
0EFE: A0 05    >19           ldy   #5          ; Copy MEM operand to rD.
0F00: B1 CA    >20   :mem2rD lda   (memptr),y
0F02: 99 AA 00 >21           sta   rD,y
0F05: 88       >22           dey
0F06: D0 F8    >23           bne   :mem2rD     ; (rD sign already set)
0F08: 84 D0    >24           sty   t1          ; Init t1 = 0
0F0A: A2 01    >25           ldx   #EXP        ; Compare rA & rD magnitudes
0F0C: B5 9E    >26   :complp lda   rA,x
0F0E: D5 AA    >27           cmp   rD,x
0F10: 90 3B    >28           bcc   :Alt        ; rA < rD.
0F12: D0 05    >29           bne   :Age        ; rA > rD.
0F14: E8       >30           inx               ; rA = rD so far...
0F15: E0 06    >31           cpx   #6
0F17: D0 F3    >32           bne   :complp
0F19: F8       >33   :Age    sed               ; / Decimal mode.
0F1A: A5 9F    >34           lda   rA+EXP      ; rA >= rD. C = 1.
0F1C: E5 AB    >35           sbc   rD+EXP      ; Operand misalignment
0F1E: F0 3D    >36           beq   :doarith    ; Misalignment = 0, go.
0F20: C9 08    >37           cmp   #8          ; Is misalignment > 7?
0F22: B0 7E    >38           bcs   :done       ; -Yes, rA unchanged.
0F24: 4A       >39           lsr               ; -No, div by 2, C = odd.
0F25: 90 0E    >40           bcc   :bytesh     ; Even, so shift bytes.
0F27: A2 04    >41           ldx   #4          ; Odd.  4 bits / digit.
0F29: 18       >42   :digsh  clc               ; Shift rD right 1 digit.
0F2A: 66 AC    >43           ror   rD+MANT
0F2C: 66 AD    >44           ror   rD+MANT+1
0F2E: 66 AE    >45           ror   rD+MANT+2
0F30: 66 AF    >46           ror   rD+MANT+3
0F32: CA       >47           dex
0F33: D0 F4    >48           bne   :digsh
0F35: A8       >49   :bytesh tay               ; Byte shift count
0F36: F0 25    >50           beq   :doarith    ; -Ready to go.
0F38: A5 AE    >51   :bytenxt lda  rD+MANT+2   ; -Shift right 2 digits
0F3A: 85 AF    >52           sta   rD+MANT+3
0F3C: A5 AD    >53           lda   rD+MANT+1
0F3E: 85 AE    >54           sta   rD+MANT+2
0F40: A5 AC    >55           lda   rD+MANT
0F42: 85 AD    >56           sta   rD+MANT+1
0F44: A9 00    >57           lda   #0
0F46: 85 AC    >58           sta   rD+MANT
0F48: 88       >59           dey
0F49: D0 ED    >60           bne   :bytenxt
0F4B: F0 10    >61           beq   :doarith    ; (always)
               >62
0F4D: A2 05    >63   :Alt    ldx   #5          ; Exchange rA and rD
0F4F: B5 9E    >64   :exchAD lda   rA,x        ;  so |rA| > |rD|.
0F51: B4 AA    >65           ldy   rD,x
0F53: 94 9E    >66           sty   rA,x
```

```
0F55: 95 AA    >67             sta   rD,x
0F57: CA       >68             dex
0F58: 10 F5    >69             bpl   :exchAD
0F5A: 38       >70             sec              ; Now |rA| >= |rD|.
0F5B: B0 BC    >71             bcs   :Age       ; (always)
               >72
0F5D: A5 9E    >73   :doarith lda   rA+S       ; Compare signs.
0F5F: C5 AA    >74             cmp   rD+S
0F61: D0 43    >75             bne   :subtr     ; -Different, subtract.
0F63: A2 03    >76             ldx   #3         ; -Same, add.
0F65: 18       >77             clc
0F66: B5 A0    >78   :add      lda   rA+MANT,x  ; rA mantissa =
0F68: 75 AC    >79             adc   rD+MANT,x  ;  rA mantissa +
0F6A: 95 A0    >80             sta   rA+MANT,x  ;   rD mantissa.
0F6C: 05 D0    >81             ora   t1         ; Summarize zero
0F6E: 85 D0    >82             sta   t1         ;  mantissa.
0F70: CA       >83             dex
0F71: 10 F3    >84             bpl   :add
0F73: B0 06    >85             bcs   :carry     ; Carry out of mantissa.
0F75: A5 D0    >86             lda   t1         ; Result mantissa = 0?
0F77: F0 41    >87             beq   :clrexp    ; -Yes, Result = 0.
0F79: D0 43    >88             bne   :norm      ; -No, normalize. (always)
               >89
0F7B: A5 9F    >90   :carry    lda   rA+EXP     ; -Carry into EXP field.
0F7D: C9 99    >91             cmp   #$99       ; Is EXP = 99 (max)?
0F7F: D0 0A    >92             bne   :adj       ; -No, shift right.
0F81: A9 01    >93             lda   #$01       ; -Yes, force EXP
0F83: 85 9F    >94             sta   rA+EXP     ;   to 01 (unshifted sum)
0F85: A9 00    >95             lda   #0         ;    and force rA sign
0F87: 85 9E    >96             sta   rA+S       ;     to 0.
0F89: F0 13    >97             beq   :ovflo     ;      and overflow. (always)
               >98
0F8B: 38       >99   :adj      sec              ; Restore the carry out.
0F8C: A2 04    >100            ldx   #4         ; 4 bits / digit.
0F8E: 20 04 15 >101  :srloop   jsr   srAM       ; -Shift mant 1 dig right.
0F91: 18       >102            clc              ; Shift in zeroes.
0F92: CA       >103            dex
0F93: D0 F9    >104            bne   :srloop
0F95: 18       >105            clc
0F96: A5 9F    >106            lda   rA+EXP     ; Increment rA exponent.
0F98: 69 01    >107            adc   #1
0F9A: 85 9F    >108            sta   rA+EXP
0F9C: 90 04    >109            bcc   :done      ; -No overflow.
               >110  :ovflo    seti  Ov         ; -Signal exponent overflow.
0F9E: A9 FF    >110            lda   #$FF
0FA0: 85 C3    >110            sta   Ov         ; Set non-zero.
               >110            eom
0FA2: D8       >111  :done     cld              ; \ Back to binary.
0FA3: 4C 01 09 >112            jmp   fetch
               >113
0FA6: A2 03    >114  :subtr    ldx   #3         ; Subtract.
0FA8: 38       >115            sec
0FA9: B5 A0    >116  :sub      lda   rA+MANT,x  ; rA mantissa =
0FAB: F5 AC    >117            sbc   rD+MANT,x  ;  rA mantissa -
0FAD: 95 A0    >118            sta   rA+MANT,x  ;   rD mantissa.
0FAF: 05 D0    >119            ora   t1         ; Summarize zero
0FB1: 85 D0    >120            sta   t1         ;  mantissa.
0FB3: CA       >121            dex
0FB4: 10 F3    >122            bpl   :sub
0FB6: A5 D0    >123            lda   t1         ; Result mantissa = 0?
0FB8: D0 04    >124            bne   :norm      ; -No, normalize.
0FBA: 85 9F    >125  :clrexp   sta   rA+EXP     ; -Yes, exponent = 0.
0FBC: F0 E4    >126            beq   :done      ; (always)
               >127
0FBE: A5 A0    >128  :norm     lda   rA+MANT    ; Normalize result.
0FC0: 29 F0    >129            and   #$F0       ; Hi digit = 0?
0FC2: D0 DE    >130            bne   :done      ; -No, all done.
```

```
                              ===== Page 65 =====

0FC4: A2 04    >131           ldx   #4          ; -Yes, shift left 1 dig.
0FC6: 18       >132  :diglp   clc               ; Shift in zeroes.
0FC7: 26 A3    >133           rol   rA+MANT+3
0FC9: 26 A2    >134           rol   rA+MANT+2
0FCB: 26 A1    >135           rol   rA+MANT+1
0FCD: 26 A0    >136           rol   rA+MANT
0FCF: CA       >137           dex
0FD0: D0 F4    >138           bne   :diglp
0FD2: C6 D1    >139           dec   NN          ; Norm limit exceeded?
0FD4: 10 04    >140           bpl   :ok         ; -No, continue.
               >141           resi  RUN         ; -Limit exceeded, halt.
0FD6: A9 00    >141           lda   #0
0FD8: 85 C0    >141           sta   RUN         ; Zero indicator.
               >141           eom
0FDA: 38       >142  :ok      sec
0FDB: A5 9F    >143           lda   rA+EXP      ; Decrement rA exponent
0FDD: E9 01    >144           sbc   #1
0FDF: 85 9F    >145           sta   rA+EXP
0FE1: B0 DB    >146           bcs   :norm
0FE3: A2 9E    >147           ldx   #rA         ; Exponent underflow,
0FE5: 20 68 15 >148           jsr   clear       ;  clear rA.
0FE8: 4C A2 0F >149           jmp   :done
               >150
0FEB: 29 01    >151  FSU      and   #$01        ; Standardize sign of
0FED: 85 AA    >152           sta   rD+S        ;  MEM operand (0/1).
0FEF: A5 9A    >153           lda   rC+VV       ; FSU or FSA?
0FF1: 29 0F    >154           and   #$0F
0FF3: C9 01    >155           cmp   #1
0FF5: F0 04    >156           beq   :setneg     ; -FSA, set operand -.
0FF7: A5 AA    >157           lda   rD+S        ; -FSU.
0FF9: 49 01    >158           eor   #$01        ;   Complement sign
0FFB: 85 AA    >159  :setneg  sta   rD+S        ;    of operand,
0FFD: 4C EC 0E >160           jmp   ]fad        ;     and do FAD.
```

===== Page 66 =====

```
1000: 18       >162  FMU      clc              ; Floating MUltiply
1001: C8       >163           iny              ; Y = 1 (exponent field)
1002: F8       >164           sed              ; / Decimal mode.
1003: B1 CA    >165           lda   (memptr),y ; Operand exponent
1005: 85 CC    >166           sta   ptr        ; Save for restoration.
1007: 65 9F    >167           adc   rA+EXP     ;  + rA exponent
1009: 90 0A    >168           bcc   :notov     ; No overflow.
100B: C9 50    >169           cmp   #$50       ; Sum < 150?
100D: 90 0A    >170           bcc   :ok        ; -Yes, no overflow.
100F: A9 80    >171           lda   #$80       ; -No, signal pending
1011: 85 C3    >172           sta   Ov         ;   FMU overflow
1013: B0 09    >173           bcs   :cont      ;    and continue a bit.
               >174
1015: C9 50    >175  :notov   cmp   #$50       ; Sum < 50?
1017: 90 71    >176           bcc   :unflow    ; -Yes, underflow.
1019: 38       >177  :ok      sec              ; -No, subtract extra
101A: E9 50    >178           sbc   #$50       ;   excess 50 and
101C: 85 D1    >179           sta   NN         ;    save result exponent.
101E: A9 00    >180  :cont    lda   #0         ; Clear operand and
1020: 91 CA    >181           sta   (memptr),y ;  rA exponents.
1022: 85 9F    >182           sta   rA+EXP
1024: A5 A0    >183           lda   rA+MANT    ; Is rA unnormalized?
1026: 29 F0    >184           and   #$F0
1028: F0 60    >185           beq   :unflow    ; -Yes, underflow.
102A: C8       >186           iny              ; Y = 2 (mantissa)
102B: B1 CA    >187           lda   (memptr),y ; Is memory operand
102D: 29 F0    >188           and   #$F0       ;  unnormalized?
102F: F0 59    >189           beq   :unflow    ; -Yes, underflow.
1031: A5 AA    >190           lda   rD+S       ; Recover operand sign.
1033: 20 D8 0C >191           jsr   multiply   ; Do the multiply.
1036: A5 C3    >192           lda   Ov         ; FMU overflow pending?
1038: C9 80    >193           cmp   #$80
103A: F0 47    >194           beq   :ovflow    ; -Yes, quit.
103C: A2 02    >195           ldx   #2         ; -No, shift rA & rR
103E: B5 9F    >196  :shloop  lda   rA+1,x     ;  left one byte.
1040: 95 9E    >197           sta   rA,x
1042: E8       >198           inx
1043: E0 06    >199           cpx   #6         ; Skip rR sign byte.
1045: D0 05    >200           bne   :notsign
1047: A5 A5    >201           lda   rR+1
1049: 85 A3    >202           sta   rA+5
104B: E8       >203           inx
104C: E0 0B    >204  :notsign cpx   #11        ; Done?
104E: D0 EE    >205           bne   :shloop    ; -No, continue.
1050: A9 00    >206           lda   #0         ; -Yes, clear
1052: 85 A9    >207           sta   rR+5       ;   low byte of rR.
1054: A5 A0    >208           lda   rA+MANT    ; Is rA normalized?
1056: 29 F0    >209           and   #$F0
1058: D0 13    >210           bne   :normal    ; -Yes.
105A: A0 04    >211           ldy   #4         ; -No, shift rA & rR
105C: 18       >212  :shdig   clc              ;   left one digit.
105D: 20 31 15 >213           jsr   slT
1060: 88       >214           dey
1061: D0 F9    >215           bne   :shdig
1063: A5 D1    >216           lda   NN         ; Recover result exp
1065: F0 23    >217           beq   :unflow    ; Underflow if 0.
1067: F8       >218           sed              ; / Decimal mode.
1068: 38       >219           sec
1069: E9 01    >220           sbc   #1         ; Compensate for shift.
106B: 85 D1    >221           sta   NN
106D: A5 D1    >222  :normal  lda   NN
106F: 85 9F    >223           sta   rA+EXP     ; Set result exponent.
1071: D8       >224  :done    cld              ; \ Binary mode.
1072: A5 C3    >225           lda   Ov         ; Pending FMU ovflow?
1074: F0 04    >226           beq   :noOv      ; -No.
               >227           seti  Ov         ; -Yes, standardize it.
1076: A9 FF    >227           lda   #$FF
```

```
1078: 85 C3    >227           sta   Ov          ; Set non-zero.
               >227           eom
107A: A0 01    >228  :noOv    ldy   #1          ; Restore memory
107C: A5 CC    >229           lda   ptr         ;  operand's exponent.
107E: 91 CA    >230           sta   (memptr),y
1080: 4C 01 09 >231           jmp   fetch
               >232
1083: A9 00    >233  :ovflow  lda   #0
1085: 85 A4    >234           sta   rR+S        ; Clear rR sign
1087: 4C 71 10 >235           jmp   :done       ;  and clean up.
               >236
108A: 20 90 10 >237  :unflow  jsr   clearAR     ; Clear rA and rR
108D: 4C 71 10 >238           jmp   :done       ;  and clean up.
               >239
1090: A2 9E    >240  clearAR  ldx   #rA         ; Clear rA.
1092: 20 68 15 >241           jsr   clear
1095: A2 A4    >242           ldx   #rR         ; Clear rR.
1097: 20 68 15 >243           jsr   clear
109A: 60       >244           rts
```

```
109B: C8       >246  FDV      iny              ; Floating DiVide (Y==>EXP)
109C: B1 CA    >247           lda   (memptr),y ; Save MEM exponent
109E: 85 CC    >248           sta   ptr        ;  for restoration
10A0: A9 00    >249           lda   #0         ;   and clear it for
10A2: 91 CA    >250           sta   (memptr),y ;    for divide.
10A4: C8       >251           iny              ; Y ==> MEM mantissa
10A5: B1 CA    >252           lda   (memptr),y ; Hi byte of mant
10A7: 29 F0    >253           and   #$F0       ; Divisor normalized?
10A9: F0 5D    >254           beq   :denorm    ; -No, overflow.
10AB: A5 A0    >255           lda   rA+MANT    ; Hi byte of rA mant
10AD: 29 F0    >256           and   #$F0       ; Dividend normalized?
10AF: F0 67    >257           beq   :unflo     ; -No, underflow.
10B1: F8       >258           sed              ; /Decimal mode.
10B2: 38       >259           sec
10B3: A5 9F    >260           lda   rA+EXP     ; Dividend exponent
10B5: E5 CC    >261           sbc   ptr        ;  - divisor exponent.
10B7: B0 07    >262           bcs   :chkov     ; *dend >= *isor, ck ovflo.
10B9: 38       >263           sec              ; *dend <  *isor, ck unflo.
10BA: E9 50    >264           sbc   #$50       ; Restore excess-50
10BC: 90 5A    >265           bcc   :unflo     ; Exponent underflow.
10BE: B0 05    >266           bcs   :ok        ; (always)
               >267
10C0: 18       >268  :chkov   clc
10C1: 69 50    >269           adc   #$50       ; Restore excess-50
10C3: B0 3F    >270           bcs   :ovflo     ; Exponent overflow.
10C5: 85 D1    >271  :ok      sta   NN         ; Save result exponent.
10C7: A9 00    >272           lda   #0         ; Clear rA exponent
10C9: 85 9F    >273           sta   rA+EXP     ;  for divide.
10CB: A0 04    >274           ldy   #4         ; 4 bits/digit.
10CD: 18       >275  :shrt    clc              ; Shift in zeros.
10CE: 20 0F 15 >276           jsr   srAMR      ; Shift rA mant & rR
10D1: 88       >277           dey              ;  right one digit.
10D2: D0 F9    >278           bne   :shrt
10D4: A5 A4    >279           lda   rR+S       ; Save original rR sign
10D6: 48       >280           pha
10D7: A5 AA    >281           lda   rD+S       ; Y=0, A=MEM sign
10D9: 20 61 0D >282           jsr   divide     ; Divide clears decimal mode.
10DC: 68       >283           pla              ; Restore original rR sign
10DD: 85 A4    >284           sta   rR+S
10DF: A5 9F    >285           lda   rA+1       ; Hi byte of quotient.
10E1: 29 F0    >286           and   #$F0       ; Is hi digit = 0?
10E3: D0 0C    >287           bne   :shrT2     ; -No, shift right 2 digs.
10E5: A0 04    >288           ldy   #4         ; -Yes, shift right 1 dig.
10E7: 18       >289  :shloop  clc              ; Shift in zeros.
10E8: 20 0D 15 >290           jsr   srT        ; Shift |rA| & |rR|
10EB: 88       >291           dey              ;  right one digit.
10EC: D0 F9    >292           bne   :shloop
10EE: 18       >293           clc              ; Indicate no overflow.
10EF: F0 0D    >294           beq   :setexp    ; (always)
               >295
10F1: F8       >296  :shrT2   sed              ; / Decimal mode.
10F2: 18       >297           clc
10F3: A5 D1    >298           lda   NN
10F5: 69 01    >299           adc   #1         ; EXP = EXP + 1
10F7: 85 D1    >300           sta   NN
10F9: B0 0D    >301           bcs   :denorm    ; Exponent overflow
10FB: 20 1D 15 >302           jsr   srT2       ; Make room for exponent
10FE: A5 D1    >303  :setexp  lda   NN         ; Set quotient exponent.
1100: 85 9F    >304           sta   rA+EXP
1102: 90 0A    >305           bcc   :done      ; (always)
               >306
1104: A9 00    >307  :ovflo   lda   #0         ; On exponent overflow
1106: 85 9F    >308           sta   rA+EXP     ;  clear result exponent.
1108: 85 9E    >309  :denorm  sta   rA+S       ; Clear rA sign and
               >310           seti  Ov         ;  set Overflow indicator.
110A: A9 FF    >310           lda   #$FF
110C: 85 C3    >310           sta   Ov         ; Set non-zero.
```

===== Page 69 =====

```
                >310          eom
110E: A5 CC     >311  :done    lda   ptr         ; Recover MEM exponent
1110: A0 01     >312           ldy   #1          ;  and put it back into
1112: 91 CA     >313           sta   (memptr),y ;   divisor in memory.
1114: D8        >314           cld               ; \ Binary mode.
1115: 4C 01 09 >315           jmp   fetch
                >316
1118: 20 90 10 >317  :unflo   jsr   clearAR     ; Clear rA and rR
111B: 4C 0E 11 >318           jmp   :done       ;  and finish up.
```

```
111E: A9 18    >320  IFL      lda   #CLCop     ; Patch ]dfl for IFL
1120: 8D BD 11 >321           sta   ]clc
1123: A9 65    >322           lda   #ADCZop
1125: 8D CC 11 >323           sta   ]adc
1128: A9 C9    >324           lda   #CMPIop
112A: 8D CE 11 >325           sta   ]cmp
112D: A9 EA    >326           lda   #NOPop
112F: 8D F6 11 >327           sta   ]nop
1132: A9 79    >328           lda   #ADCYop
1134: 8D F9 11 >329           sta   ]sub
1137: A9 C3    >330           lda   #Ov
1139: 8D 18 12 >331           sta   ]Ov+3
113C: 20 89 11 >332           jsr   ]dfl       ; Do the IFL.
113F: A9 C4    >333           lda   #Rp        ; Patch ]dfl back.
1141: 8D 18 12 >334           sta   ]Ov+3
1144: A9 F9    >335           lda   #SBCYop
1146: 8D F9 11 >336           sta   ]sub
1149: A9 38    >337           lda   #SECop
114B: 8D F6 11 >338           sta   ]nop
114E: A9 24    >339           lda   #BITZop
1150: 8D CE 11 >340           sta   ]cmp
1153: A9 E5    >341           lda   #SBCZop
1155: 8D CC 11 >342           sta   ]adc
1158: A9 EA    >343           lda   #NOPop
115A: 8D BD 11 >344           sta   ]clc
115D: A5 C1    >345           lda   ERR        ; Error detected?
115F: D0 10    >346           bne   ]errpt     ; -Yes, report it.
1161: 4C 01 09 >347  ]fetch4  jmp   fetch
               >348
               >349  DFL      resi  Rp         ; Reset Repeat indicator.
1164: A9 00    >349           lda   #0
1166: 85 C4    >349           sta   Rp         ; Zero indicator.
               >349           eom
1168: 20 89 11 >350           jsr   ]dfl       ; Decrease FieLd
116B: A5 C1    >351           lda   ERR        ; Error detected?
116D: D0 02    >352           bne   ]errpt     ; -Yes, report it.
116F: F0 F0    >353           beq   ]fetch4    ; (always)
               >354
1171: 4C DB 09 >355  ]errpt   jmp   ]err
               >356
               >357  DLB      resi  Rp         ; Reset Repeat indicator.
1174: A9 00    >357           lda   #0
1176: 85 C4    >357           sta   Rp         ; Zero indicator.
               >357           eom
1178: 20 89 11 >358           jsr   ]dfl       ; Decrease Field
117B: A5 AD    >359           lda   rD+3       ; Load rB from rD 8:4.
117D: 85 94    >360           sta   rB
117F: A5 AE    >361           lda   rD+4
1181: 85 95    >362           sta   rB+1
1183: A5 C1    >363           lda   ERR        ; Error detected?
1185: D0 EA    >364           bne   ]errpt     ; -Yes, report it.
1187: F0 D8    >365           beq   ]fetch4    ; (always)
```

```
1189: A2 AA    >367  ]dfl     ldx   #rD        ; Clear rD.
118B: 20 68 15 >368           jsr   clear
118E: A2 B0    >369           ldx   #rD10      ; Clear rD10.
1190: 20 68 15 >370           jsr   clear
1193: 20 54 15 >371           jsr   splitsL    ; A = s, X = L
1196: 18       >372           clc
1197: 69 01    >373           adc   #1         ; A = s + 1
1199: 4A       >374           lsr              ; A = (s+1)/2, C = even dig
119A: 08       >375           php              ; Push Carry status.
119B: A8       >376           tay              ; Y = low byte index
119C: A5 9A    >377           lda   rC+VV      ; NN
119E: 99 B0 00 >378           sta   rD10,y     ; rD10 = subtrahend
11A1: B0 16    >379           bcs   :subtr     ; Even dig first, no shift.
11A3: 86 D0    >380           stx   t1         ; Save X
11A5: 98       >381           tya              ; Move Y to X.
11A6: AA       >382           tax
11A7: 16 B0    >383           asl   rD10,x     ; Odd dig first, shift
11A9: 36 AF    >384           rol   rD10-1,x   ;  1 digit left.
11AB: 16 B0    >385           asl   rD10,x
11AD: 36 AF    >386           rol   rD10-1,x
11AF: 16 B0    >387           asl   rD10,x
11B1: 36 AF    >388           rol   rD10-1,x
11B3: 16 B0    >389           asl   rD10,x
11B5: 36 AF    >390           rol   rD10-1,x
11B7: A6 D0    >391           ldx   t1         ; Restore X.
11B9: 28       >392  :subtr   plp              ; Pop C.
11BA: F8       >393           sed              ; / Decimal mode.
11BB: 90 39    >394           bcc   ]nop       ; Not C = odd dig first.
11BD: EA       >395  ]clc     nop              ; <Patch to CLC for IFL>
11BE: CA       >396  :evendig dex              ; Both even and odd digs?
11BF: D0 36    >397           bne   :byte      ; -Yes, subtr whole byte.
11C1: B9 B0 00 >398           lda   rD10,y     ; -No, subtr final digit.
11C4: 29 0F    >399           and   #$0F       ; Isolate even digit
11C6: 85 D0    >400           sta   t1         ;  and save for subtract.
11C8: B1 CA    >401           lda   (memptr),y ; MEM byte
11CA: 29 0F    >402           and   #$0F       ; Isolate even digit
11CC: E5 D0    >403  ]adc     sbc   t1         ;  & subtr. <ADC for IFL>
11CE: 24 10    >404  ]cmp     bit   $10        ; CMP# if IFL (to set C)
11D0: 29 0F    >405           and   #$0F       ; Mask result
11D2: 85 D0    >406           sta   t1         ;  and save it.
11D4: B1 CA    >407           lda   (memptr),y ; Recover MEM byte,
11D6: 29 F0    >408           and   #$F0       ;  mask out even digit,
11D8: 05 D0    >409           ora   t1         ;   OR in difference,
11DA: 91 CA    >410           sta   (memptr),y ;    and put it back.
11DC: A4 AE    >411           ldy   rD+4       ; Save high 4 digits of
11DE: 84 AF    >412           sty   rD+5       ;  difference in rD 8:4.
11E0: A4 AD    >413           ldy   rD+3
11E2: 84 AE    >414           sty   rD+4
11E4: 85 AD    >415           sta   rD+3
11E6: 08       >416           php              ; Push Carry status.
11E7: A2 04    >417           ldx   #4         ; 4 bits/digit
11E9: 26 AF    >418  :shlp    rol   rD+5       ; Shift rD left 1 digit
11EB: 26 AE    >419           rol   rD+4       ;  to line up with rB.
11ED: 26 AD    >420           rol   rD+3
11EF: CA       >421           dex
11F0: D0 F7    >422           bne   :shlp
11F2: 28       >423           plp              ; Pop Carry status.
11F3: 4C 12 12 >424           jmp   :done
               >425
11F6: 38       >426  ]nop     sec              ; <Patch to NOP for IFL>
11F7: B1 CA    >427  :byte    lda   (memptr),y ; MEM byte
11F9: F9 B0 00 >428  ]sub     sbc   rD10,y     ;  minus subtrahend
11FC: 91 CA    >429           sta   (memptr),y ;   back to MEM.
11FE: 84 D0    >430           sty   t1         ; Save Y
1200: A4 AE    >431           ldy   rD+4       ; Save 4 hi digits of
1202: 84 AF    >432           sty   rD+5       ;  difference in rD 8:4.
1204: A4 AD    >433           ldy   rD+3
```

```
1206: 84 AE    >434              sty   rD+4
1208: 85 AD    >435              sta   rD+3
120A: A4 D0    >436              ldy   t1          ; Restore Y
120C: 88       >437              dey
120D: 30 0B    >438              bmi   :flderr     ; Field error.
120F: CA       >439              dex               ; More digits?
1210: D0 AC    >440              bne   :evendig    ; -Yes, keep subtracting.
1212: D8       >441  :done       cld               ; \ -No. Back to binary.
1213: 90 04    >442              bcc   :noRpt      ; Underflow ==> no Rpt
               >443  ]Ov         seti  Rp          ; Set Rpt <Ov for IFL>
1215: A9 FF    >443              lda   #$FF
1217: 85 C4    >443              sta   Rp          ; Set non-zero.
               >443              eom
1219: 60       >444  :noRpt      rts
               >445
121A: A9 C6    >446  :flderr     lda   #"F"        ; Signal Field error
121C: 85 C1    >447              sta   ERR
121E: D8       >448              cld               ; Clear decimal mode.
121F: 60       >449              rts
```

```
1220: 84 CF    >451  RTF      sty   inptr+1    ; 'inptr+1' = 0
1222: 84 D0    >452           sty   t1         ; 't1' = 0
1224: 20 75 15 >453           jsr   midNN      ; Extract NN (word count)
1227: 85 CE    >454           sta   inptr      ; Save binary NN (1..100)
1229: A6 95    >455           ldx   rB+1       ; Convert rB to MEM
122B: E0 9A    >456           cpx   #$99+1     ;  address in 'ptr'.
122D: B0 51    >457           bcs   :underr    ; Undigit error.
122F: A4 94    >458           ldy   rB
1231: C0 4A    >459           cpy   #$49+1
1233: B0 4E    >460           bcs   :addrerr   ; Address error.
1235: BD C7 19 >461           lda   BCDLadrl,x
1238: 79 FB 1A >462           adc   BCDHadrl,y
123B: 85 CC    >463           sta   ptr
123D: BD 61 1A >464           lda   BCDLadrh,x
1240: 79 45 1B >465           adc   BCDHadrh,y
1243: B0 3B    >466           bcs   :underr    ; Carry out ==> undigit.
1245: 85 CD    >467           sta   ptr+1      ; 'ptr' = dest MEM addr.
1247: A5 CE    >468           lda   inptr      ; Binary NN
1249: 0A       >469           asl              ; NN * 2 (2..200)
124A: 65 CE    >470           adc   inptr      ; NN * 3 (3..300)
124C: 26 CF    >471           rol   inptr+1    ; Capture high bit.
124E: 0A       >472           asl
124F: 26 CF    >473           rol   inptr+1    ; NN * 6 (6..600)
1251: AA       >474           tax              ; Byte count lo
1252: A0 00    >475           ldy   #0
1254: B1 CA    >476  :movelp  lda   (memptr),y ; Move bytes upward.
1256: 91 CC    >477           sta   (ptr),y
1258: CA       >478           dex              ; Dec byte count lo
1259: F0 09    >479           beq   :ckhi      ; If 0, chk hi byte.
125B: C8       >480  :cont    iny
125C: D0 F6    >481           bne   :movelp
125E: E6 CB    >482           inc   memptr+1   ; Advance ptr pages
1260: E6 CD    >483           inc   ptr+1
1262: D0 F0    >484           bne   :movelp    ; (always)
               >485
1264: C6 CF    >486  :ckhi    dec   inptr+1    ; Dec byte count hi
1266: 10 F3    >487           bpl   :cont      ; Continue if >= 0.
1268: A5 D1    >488           lda   NN         ; NN = 00 (100)?
126A: D0 02    >489           bne   :lt100     ; -No, less than 100.
126C: E6 D0    >490           inc   t1         ; -Yes, set 100.
126E: F8       >491  :lt100   sed              ; / Decimal mode.
126F: 18       >492           clc
1270: A5 95    >493           lda   rB+1       ; rB = rB + NN
1272: 65 D1    >494           adc   NN
1274: 85 95    >495           sta   rB+1
1276: A5 94    >496           lda   rB
1278: 65 D0    >497           adc   t1         ; 1 if NN = 0, else 0.
127A: 85 94    >498           sta   rB
127C: D8       >499           cld              ; \ Back to binary.
127D: 4C 01 09 >500           jmp   fetch
               >501
1280: 4C D9 09 >502  :underr  jmp   UNDIGerr   ; Relay jump.
1283: 4C CF 09 >503  :addrerr jmp   ADDRerr    ; Relay jump.
```

===== Page 74 =====

```
1286: F8       >505  IBB      sed              ; / Decimal mode.
1287: 18       >506           clc
1288: A5 95    >507           lda   rB+1       ; rB = rB + rC(4:4)
128A: 65 9A    >508           adc   rC+VV
128C: 85 95    >509           sta   rB+1
128E: A5 94    >510           lda   rB
1290: 65 99    >511           adc   rC+sL
1292: 85 94    >512           sta   rB
1294: D8       >513           cld              ; \ Back to binary.
1295: 90 58    >514           bcc   BUN        ; No overflow ==> branch
1297: B0 66    >515           bcs   ]fetch3    ; Overflow ==> continue
               >516
1299: F8       >517  DBB      sed              ; / Decimal mode.
129A: 38       >518           sec
129B: A5 95    >519           lda   rB+1       ; rB = rB - rC(4:4)
129D: E5 9A    >520           sbc   rC+VV
129F: 85 95    >521           sta   rB+1
12A1: A5 94    >522           lda   rB
12A3: E5 99    >523           sbc   rC+sL
12A5: 85 94    >524           sta   rB
12A7: D8       >525           cld              ; \ Back to binary.
12A8: B0 45    >526           bcs   BUN        ; No underflow ==> branch
12AA: 90 53    >527           bcc   ]fetch3    ; Underflow. (always)
```

```
12AC: A5 C3    >529  BOF      lda   Ov          ; Overflow indicator set?
12AE: D0 02    >530           bne   :ovflo      ; -Yes, clear it and branch.
12B0: F0 4D    >531           beq   ]fetch3     ; (always)
               >532
               >533  :ovflo   resi  Ov          ; Reset Overflow indicator
12B2: A9 00    >533           lda   #0
12B4: 85 C3    >533           sta   Ov          ; Zero indicator.
               >533           eom
12B6: 4C EF 12 >534           jmp   BUN         ;  and take the branch.
               >535
12B9: A5 C4    >536  BRP      lda   Rp          ; Repeat indicator set?
12BB: D0 32    >537           bne   BUN         ; -Yes, branch.
12BD: F0 40    >538           beq   ]fetch3     ; (always)
               >539
12BF: A5 9A    >540  BSA      lda   rC+VV       ; Get comparand digit
12C1: 29 0F    >541           and   #$0F
12C3: C5 9E    >542           cmp   rA+S        ; Equal to rA sign?
12C5: F0 28    >543           beq   BUN         ; -Yes, take branch.
12C7: D0 36    >544           bne   ]fetch3     ; (always)
               >545
12C9: A5 9A    >546  BCH      lda   rC+VV       ; BCH or BCL?
12CB: 29 01    >547           and   #$01
12CD: F0 06    >548           beq   :bch        ; -BCH.
12CF: A5 C2    >549           lda   COMP        ; -BCL.
12D1: 30 1C    >550           bmi   BUN         ; Branch if Lo
12D3: 10 2A    >551           bpl   ]fetch3     ; (always)
               >552
12D5: A5 C2    >553  :bch     lda   COMP
12D7: F0 26    >554           beq   ]fetch3     ; Equal.
12D9: 10 14    >555           bpl   BUN         ; Branch if Hi
12DB: 30 22    >556           bmi   ]fetch3     ; (always)
               >557
12DD: A5 9A    >558  BCE      lda   rC+VV       ; BCE or BCU?
12DF: 29 01    >559           and   #$01
12E1: F0 06    >560           beq   :bce        ; BCE.
12E3: A5 C2    >561           lda   COMP
12E5: D0 08    >562           bne   BUN         ; Branch if unequal.
12E7: F0 16    >563           beq   ]fetch3     ; (always)
               >564
12E9: A5 C2    >565  :bce     lda   COMP
12EB: F0 02    >566           beq   BUN         ; Branch if equal.
12ED: D0 10    >567           bne   ]fetch3     ; (always)
               >568
12EF: A5 9C    >569  BUN      lda   rC+ADDR     ; Set new P reg
12F1: 85 96    >570           sta   rP
12F3: A5 9D    >571           lda   rC+ADDR+1
12F5: 85 97    >572           sta   rP+1
12F7: A5 CA    >573           lda   memptr      ;  and instptr.
12F9: 85 C8    >574           sta   instptr
12FB: A5 CB    >575           lda   memptr+1
12FD: 85 C9    >576           sta   instptr+1
12FF: 4C 01 09 >577  ]fetch3  jmp   fetch
```

```
1302: A2 A4    >579  BFR      ldx   #rR        ; X points to rR
1304: D0 02    >580           bne   ]bfr
               >581
1306: A2 9E    >582  BFA      ldx   #rA        ; X points to rA
1308: A4 9A    >583  ]bfr     ldy   rC+VV      ; Y = 2-digit comparand
130A: A5 99    >584           lda   rC+sL
130C: 29 10    >585           and   #$10       ; s even or odd?
130E: F0 0E    >586           beq   :even      ; -Even, no digit swap.
1310: 98       >587           tya              ; -Odd, swap digits.
1311: C9 80    >588           cmp   #$80       ; Hi bit to C
1313: 2A       >589           rol              ;  and rotate 1 bit.
1314: C9 80    >590           cmp   #$80       ; Hi bit to C
1316: 2A       >591           rol              ;  and rotate 1 bit.
1317: C9 80    >592           cmp   #$80       ; Hi bit to C
1319: 2A       >593           rol              ;  and rotate 1 bit.
131A: C9 80    >594           cmp   #$80       ; Hi bit to C
131C: 2A       >595           rol              ;  and rotate 1 bit.
131D: A8       >596           tay
131E: 84 B5    >597  :even    sty   rD10+5     ; Expand comparand
1320: 84 B4    >598           sty   rD10+4     ;  to full width in rD10.
1322: 84 B3    >599           sty   rD10+3
1324: 84 B2    >600           sty   rD10+2
1326: 84 B1    >601           sty   rD10+1
1328: 98       >602           tya
1329: 29 0F    >603           and   #$0F       ; Mask off hi sign digit.
132B: 85 B0    >604           sta   rD10
132D: A5 CB    >605           lda   memptr+1   ; Push 'memptr' on stack.
132F: 48       >606           pha
1330: A5 CA    >607           lda   memptr
1332: 48       >608           pha
1333: A9 B0    >609           lda   #rD10      ; Point 'memptr' at rD10
1335: 85 CA    >610           sta   memptr
1337: A9 00    >611           lda   #0
1339: 85 CB    >612           sta   memptr+1
               >613
133B: A0 01    >614           ldy   #1         ; Partial field compare
133D: A9 B0    >615           lda   #BCSop     ; Unsigned compare
133F: 20 40 0E >616           jsr   compare
1342: AA       >617           tax              ; Save A
1343: 68       >618           pla              ; Pop 'memptr'
1344: 85 CA    >619           sta   memptr
1346: 68       >620           pla
1347: 85 CB    >621           sta   memptr+1
1349: A5 C1    >622           lda   ERR        ; Error detected?
134B: D0 05    >623           bne   :err       ; -Yes, report it.
134D: 8A       >624           txa              ; Recover COMP flags
134E: F0 9F    >625           beq   BUN        ; -Branch if equal.
1350: D0 6E    >626           bne   ]fetch2    ; -Else NOP. (always)
               >627
1352: 4C DB 09 >628  :err     jmp   ]err
               >629
1355: A5 99    >630  BCS      lda   rC+sL      ; Get switch #
1357: 4A       >631           lsr
1358: 4A       >632           lsr
1359: 4A       >633           lsr
135A: 4A       >634           lsr
135B: AA       >635           tax
135C: B5 B6    >636           lda   CSW,x      ; Get switch state
135E: D0 8F    >637           bne   BUN        ; -True, take branch.
1360: F0 5E    >638           beq   ]fetch2    ; -False, no branch.
```

```
1362: A5 9A    >640  SOR      lda   rC+VV      ; SOR / SOH / IOM?
1364: 29 0F    >641           and   #$0F
1366: C9 02    >642           cmp   #2         ; IOM?
1368: F0 05    >643           beq   :iom       ; -Yes.
136A: 85 C7    >644           sta   OvHlt      ; -No, set Ovflo mode.
136C: 4C 01 09 >645  :fetch   jmp   fetch
               >646
136F: A5 C7    >647  :iom     lda   OvHlt
1371: F0 F9    >648           beq   :fetch     ; No branch if SOR mode.
1373: 4C EF 12 >649           jmp   BUN        ; Branch if SOH mode.
               >650
1376: A5 9A    >651  STA      lda   rC+VV      ; STA, STR, STB?
1378: 29 0F    >652           and   #$0F       ; Isolate reg variant.
137A: A2 A4    >653           ldx   #rR
137C: C9 01    >654           cmp   #1         ; STR?
137E: F0 08    >655           beq   :store     ; -Yes.
1380: A2 90    >656           ldx   #rBx
1382: C9 02    >657           cmp   #2         ; STB?
1384: F0 02    >658           beq   :store     ; -Yes.
1386: A2 9E    >659           ldx   #rA        ; STA
1388: A5 9A    >660  :store   lda   rC+VV      ; Partial field :store?
138A: 29 10    >661           and   #$10
138C: D0 0F    >662           bne   :stfield   ; -Yes, do it.
138E: 8E 94 13 >663           stx   :stloop+1  ; -No, full word store.
1391: A0 05    >664           ldy   #5
1393: B9 00 00 >665  :stloop  lda   0*0,y      ; Store the register.
1396: 91 CA    >666           sta   (memptr),y
1398: 88       >667           dey
1399: 10 F8    >668           bpl   :stloop
139B: 30 23    >669           bmi   ]fetch2    ; (always)
               >670
139D: 8E AE 13 >671  :stfield stx   :evendig+1 ; Save register
13A0: 8E C4 13 >672           stx   :odddig+1  ;  address...
13A3: 20 54 15 >673           jsr   splitsL    ; Split sL: A = s and X = L
13A6: 18       >674           clc
13A7: 69 01    >675           adc   #1         ; A = s + 1
13A9: 4A       >676           lsr              ; A = (s+1)/2, C = even dig
13AA: A8       >677           tay              ; Y = byte offset
13AB: 90 16    >678           bcc   :odddig    ; -Start digit is odd.
13AD: B9 00 00 >679  :evendig lda   0*0,y      ; -Start digit is even.
13B0: CA       >680           dex              ; Both even & odd digits?
13B1: D0 1D    >681           bne   :byte      ; -Yes, move full byte.
13B3: E8       >682           inx              ; -No, restore dig counter.
13B4: 29 0F    >683           and   #$0F       ; Isolate even digit
13B6: 85 D0    >684           sta   t1         ;  and save it.
13B8: B1 CA    >685           lda   (memptr),y ; Get MEM byte,
13BA: 29 F0    >686           and   #$F0       ;  clear target digit,
13BC: 05 D0    >687           ora   t1         ;   OR in new digit,
13BE: 91 CA    >688           sta   (memptr),y ;    and put it back.
13C0: 4C 01 09 >689  ]fetch2  jmp   fetch      ; All done.
               >690
13C3: B9 00 00 >691  :odddig  lda   0*0,y      ; Start digit is odd.
13C6: 29 F0    >692           and   #$F0       ; Isolate reg digit
13C8: 85 D0    >693           sta   t1         ;  and save it.
13CA: B1 CA    >694           lda   (memptr),y ; Get MEM byte,
13CC: 29 0F    >695           and   #$0F       ;  clear target digit,
13CE: 05 D0    >696           ora   t1         ;   OR in new digit,
13D0: 91 CA    >697  :byte    sta   (memptr),y ;    and put it back.
13D2: 88       >698           dey              ; Move byte index.
13D3: 30 05    >699           bmi   :flderr    ; -Err if field too long.
13D5: CA       >700           dex              ; More digits?
13D6: D0 D5    >701           bne   :evendig   ; -Yes, continue.
13D8: F0 E6    >702           beq   ]fetch2    ; -No, finished. (always)
               >703
13DA: 4C CB 09 >704  :flderr  jmp   FIELDerr   ; Report field error.
```

```
13DD: A0 05    >706  LDR      ldy   #5          ; MEM(ADDR) ==> rR
13DF: B1 CA    >707  :ldr     lda   (memptr),y
13E1: 99 A4 00 >708           sta   rR,y
13E4: 88       >709           dey
13E5: 10 F8    >710           bpl   :ldr
13E7: 30 41    >711           bmi   ]fetch1    ; (always)
               >712
13E9: A5 9A    >713  LDB      lda   rC+VV      ; LDB, LBC
13EB: A0 05    >714           ldy   #5
13ED: 29 01    >715           and   #$01
13EF: D0 0C    >716           bne   :lbc       ; Load rB Complement
13F1: B1 CA    >717  :ldb     lda   (memptr),y
13F3: 85 95    >718           sta   rB+1
13F5: 88       >719           dey
13F6: B1 CA    >720           lda   (memptr),y
13F8: 85 94    >721           sta   rB
13FA: 4C 01 09 >722           jmp   fetch      ; -Yes, done.
               >723
13FD: F8       >724  :lbc     sed              ; / Decimal mode
13FE: 38       >725           sec              ;  for 10's complement.
13FF: A9 00    >726  :ldbc    lda   #0
1401: F1 CA    >727           sbc   (memptr),y
1403: 85 95    >728           sta   rB+1
1405: 88       >729           dey
1406: A9 00    >730           lda   #0
1408: F1 CA    >731           sbc   (memptr),y
140A: 85 94    >732           sta   rB
140C: D8       >733           cld              ; \ -Yes, back to binary.
140D: 90 1B    >734           bcc   ]fetch1    ; (always)
               >735
140F: A5 9A    >736  LSA      lda   rC+VV      ; Load Sign A
1411: 29 0F    >737           and   #$0F       ; Isolate new sign digit
1413: 85 9E    >738           sta   rA+S       ;  and put into rA.
1415: 4C 01 09 >739           jmp   fetch
```

```
1418: A0 05    >741  STP      ldy   #5         ; rP + 1 ==> MEM(0:4)
141A: F8       >742           sed              ; / Decimal mode
141B: 18       >743           clc
141C: A5 97    >744           lda   rP+1
141E: 69 01    >745           adc   #1
1420: 91 CA    >746           sta   (memptr),y
1422: 88       >747           dey
1423: A5 96    >748           lda   rP
1425: 69 00    >749           adc   #0
1427: 91 CA    >750           sta   (memptr),y
1429: D8       >751           cld              ; \ Back to binary
142A: 4C 01 09 >752  ]fetch1  jmp   fetch      ; -Yes, done.
               >753
142D: A5 9A    >754  CLA      lda   rC+VV      ; CLA/R/B
142F: 4A       >755           lsr              ; 1-bit to C
1430: 85 D0    >756           sta   t1         ; Save mask
1432: 90 05    >757           bcc   :notA      ; rA not included.
1434: A2 9E    >758           ldx   #rA
1436: 20 68 15 >759           jsr   clear      ; Clear rA.
1439: 46 D0    >760  :notA    lsr   t1         ; 2-bit to C
143B: 90 05    >761           bcc   :notR      ; rR not included.
143D: A2 A4    >762           ldx   #rR
143F: 20 68 15 >763           jsr   clear      ; Clear rR.
1442: 46 D0    >764  :notR    lsr   t1         ; 4-bit to C.
1444: 90 05    >765           bcc   :fetch     ; rB not included.
1446: A2 90    >766           ldx   #rBx
1448: 20 68 15 >767           jsr   clear      ; Clear rB.
144B: 4C 01 09 >768  :fetch   jmp   fetch
               >769
144E: A9 00    >770  CLL      lda   #0         ; CLear Location
1450: A0 05    >771           ldy   #5
1452: 91 CA    >772  :cllloop sta   (memptr),y
1454: 88       >773           dey
1455: 10 FB    >774           bpl   :cllloop
1457: 30 D1    >775           bmi   ]fetch1    ; (always)
```

```
1459: A5 9D    >777  SRA      lda   rC+ADDR+1  ; SRA, SRT, SRS nn
145B: 29 1F    >778           and   #$1F       ; Isolate count 0..19
145D: C9 10    >779           cmp   #$10       ; Greater than 9?
145F: 90 02    >780           bcc   :nocor     ; -No, don't correct.
1461: E9 06    >781           sbc   #6         ; -Yes, cnvrt to binary.
1463: 0A       >782  :nocor   asl              ; Multiply digit shift
1464: 0A       >783           asl              ;  count by 4 (bits/digit).
1465: A8       >784           tay              ; Y = bit shift count.
1466: A5 9A    >785           lda   rC+VV      ; SRA, SRT, SRS
1468: 29 0F    >786           and   #$0F
146A: C9 01    >787           cmp   #1         ; SRT?
146C: D0 08    >788           bne   :notsrt    ; -No.
146E: A6 9E    >789           ldx   rA+S       ; -Yes, SRT. Set rR sign
1470: 86 A4    >790           stx   rR+S       ;   to rA sign, then
1472: A2 0D    >791           ldx   #<srT      ;    shift both A and R.
1474: D0 08    >792           bne   :setsh     ; Go shift. (always)
               >793
1476: A2 00    >794  :notsrt  ldx   #<srAS
1478: C9 02    >795           cmp   #2         ; SRS?
147A: F0 02    >796           beq   :setsh     ; -Yes, shift right A & Sign
147C: A2 02    >797           ldx   #<srA      ; SRA
147E: 8E 86 14 >798  :setsh   stx   :shiftr+1  ; Set shift subroutine.
1481: 98       >799           tya              ; Is shift count = 0?
1482: F0 07    >800           beq   :fetch     ; -Yes, done.
1484: 18       >801  :nxbit   clc              ; Shift in zeros.
1485: 20 02 15 >802  :shiftr  jsr   srA        ; (or srT or srAS)
1488: 88       >803           dey              ; Count exhausted?
1489: D0 F9    >804           bne   :nxbit     ; -No, keep shifting.
148B: 4C 01 09 >805  :fetch   jmp   fetch      ; -Yes, done.
```

===== Page 81 =====

```
148E: A5 9D    SLA      lda   rC+ADDR+1  ; SLA, SLT, SLS nn
1490: 29 1F             and   #$1F       ; Isolate count 0..19
1492: C9 10             cmp   #$10       ; Greater than 9?
1494: 90 02             bcc   :nocor     ; -No, don't correct.
1496: E9 06             sbc   #6         ; -Yes, cnvrt to binary.
1498: AA       :nocor   tax              ; X = shift count.
1499: A5 9A             lda   rC+VV      ; SLA, SLT, SLS?
149B: 29 0F             and   #$0F
149D: C9 01             cmp   #1         ; SLT?
149F: F0 19             beq   :slt       ; -Yes, shift left AR
14A1: E0 00             cpx   #0         ; -No, check count.
14A3: F0 12             beq   :fetch     ; Done if count = 0.
14A5: C9 02             cmp   #2         ; SLS?
14A7: F0 3C             beq   :sls       ; -Yes, shift left A + Sign
14A9: A0 04    :sla     ldy   #4         ; SLA. Shift 4 bits/digit.
14AB: A5 9F    :nxbita  lda   rA+1       ; To rotate rA,
14AD: 2A                rol              ;  preset C to high bit.
14AE: 20 3B 15          jsr   slA        ; Rotate A left 1 bit.
14B1: 88                dey              ; More bits?
14B2: D0 F7             bne   :nxbita    ; -Yes.
14B4: CA                dex              ; More digits?
14B5: D0 F2             bne   :sla       ; -Yes.
14B7: 4C 01 09 :fetch   jmp   fetch

14BA: A5 A4    :slt     lda   rR+S       ; Copy rR Sign
14BC: 85 9E             sta   rA+S       ;  to rA Sign.
14BE: 8A                txa              ; Is count = 0?
14BF: F0 F6             beq   :fetch     ; -Yes, done.
14C1: E0 0A             cpx   #10        ; -No, count >= 10?
14C3: 90 10             bcc   :nxdig     ; -No, do general case.
14C5: 86 D0             stx   t1         ; -Yes, special case SLT >= 10.
14C7: 20 46 15          jsr   exchAR     ; Exchange A and R magnitudes
14CA: A5 D0             lda   t1         ; Recover count.
14CC: 38                sec
14CD: E9 0A             sbc   #10        ; Is count = 10?
14CF: F0 E6             beq   :fetch     ; -Yes, done.
14D1: AA                tax              ; -No, keep shifting.
14D2: A5 9F             lda   rA+1       ; Hi magnitude digit.
14D4: 2A                rol              ; High bit to C
14D5: A0 04    :nxdig   ldy   #4         ; 4 bits/digit
14D7: A5 9F    :nxbitt  lda   rA+1       ; To rotate rA, rR
14D9: 2A                rol              ;  preset C to high bit.
14DA: 20 31 15          jsr   slT        ; Rotate AR left 1 bit.
14DD: 88                dey              ; More bits?
14DE: D0 F7             bne   :nxbitt    ; -Yes.
14E0: CA                dex              ; More digits?
14E1: D0 F2             bne   :nxdig     ; -Yes.
14E3: F0 D2             beq   :fetch     ; (always)

14E5: A0 04    :sls     ldy   #4         ; SLS. 4 bits/digit
14E7: A5 9E    :nxbit   lda   rA+S       ; Use sign digit
14E9: 29 0F             and   #$0F       ;  and mask it.
14EB: C9 08             cmp   #8         ; Hi bit of sign to C
14ED: 20 3B 15          jsr   slA        ; Rotate A left 1 bit
14F0: A5 9E             lda   rA+S       ;  then rotate sign.
14F2: 2A                rol
14F3: 29 0F             and   #$0F       ; Mask again
14F5: 85 9E             sta   rA+S       ;  and put it back.
14F7: 88                dey              ; More bits?
14F8: D0 ED             bne   :nxbit     ; -Yes.
14FA: CA                dex              ; More digits?
14FB: D0 E8             bne   :sls       ; -Yes.
14FD: F0 B8             beq   :fetch     ; (always)
```

```
               >871  ***********************************************************
               >872  *                                                         *
               >873  *            Utility Shifting Subroutines                 *
               >874  *                                                         *
               >875  ***********************************************************
               >876
               >877           align 256
14FF: 00       >877           ds    *-1/256*256+256-*
               >877           eom
               >878  ]keep    equ   */256      ; Keep here to 'kend' on one page.
               >879
1500: 66 9E    >880  srAS     ror   rA         ; rA & sign right 1 bit
1502: 66 9F    >881  srA      ror   rA+1       ; Sign not included
1504: 66 A0    >882  srAM     ror   rA+2       ; FP mantissa
1506: 66 A1    >883           ror   rA+3
1508: 66 A2    >884           ror   rA+4
150A: 66 A3    >885           ror   rA+5
150C: 60       >886           rts
               >887
150D: 66 9F    >888  srT      ror   rA+1       ; |rA| & |rR| right 1 bit
150F: 20 04 15 >889  srAMR    jsr   srAM       ; Shift rA Mantissa & |rR|
1512: 66 A5    >890  srR      ror   rR+1       ; Shift |rR|
1514: 66 A6    >891           ror   rR+2
1516: 66 A7    >892           ror   rR+3
1518: 66 A8    >893           ror   rR+4
151A: 66 A9    >894           ror   rR+5
151C: 60       >895           rts
               >896
151D: A2 0A    >897  srT2     ldx   #10        ; |rA| & |rR| right
151F: B5 9E    >898  :shloop  lda   rA,x       ;  2 digits (1 byte).
1521: E0 05    >899           cpx   #5         ; About to store in rR+S?
1523: D0 04    >900           bne   :cont      ; -No, continue.
1525: 85 A5    >901           sta   rR+1       ; -Yes, skip rR sign.
1527: F0 02    >902           beq   :next      ;   and on to next byte.
1529: 95 9F    >903  :cont    sta   rA+1,x
152B: CA       >904  :next    dex
152C: D0 F1    >905           bne   :shloop    ; Exclude rA sign.
152E: 86 9F    >906           stx   rA+1       ; Shift in zeros.
1530: 60       >907           rts
               >908
1531: 26 A9    >909  slT      rol   rR+5       ; Rotate |rR| & |rA| left
1533: 26 A8    >910           rol   rR+4       ;  one bit.
1535: 26 A7    >911           rol   rR+3
1537: 26 A6    >912           rol   rR+2
1539: 26 A5    >913           rol   rR+1       ; Fall into slA.
               >914
153B: 26 A3    >915  slA      rol   rA+5       ; Rotate |rA| left 1 bit
153D: 26 A2    >916           rol   rA+4
153F: 26 A1    >917           rol   rA+3
1541: 26 A0    >918           rol   rA+2
1543: 26 9F    >919           rol   rA+1
1545: 60       >920           rts
               >921
1546: A2 05    >922  exchAR   ldx   #5         ; Exchange |rA| and |rR|
1548: B5 9E    >923  :exch    lda   rA,x       ; (equivalent to SLT 10)
154A: B4 A4    >924           ldy   rR,x
154C: 95 A4    >925           sta   rR,x
154E: 94 9E    >926           sty   rA,x
1550: CA       >927           dex
1551: D0 F5    >928           bne   :exch
1553: 60       >929           rts
               >930
               >931  ]kend    equ   *-1/256    ; Warn if page crossing
               >932           err   ]kend-]keep ; between ]keep and ]kend.
```

```
                >934  **********************************************************
                >935  *                                                        *
                >936  *         Split sL field into A = s and X = L            *
                >937  *                                                        *
                >938  **********************************************************
                >939
1554: A5 99     >940  splitsL  lda   rC+sL      ; Get field specifier
1556: 29 0F     >941           and   #$0F       ; L = digit count
1558: D0 02     >942           bne   :notz
155A: A9 0A     >943           lda   #10        ; "0" ==> 10
155C: AA        >944  :notz    tax              ; X = digit count (L)
155D: A5 99     >945           lda   rC+sL
155F: 4A        >946           lsr              ; Isolate field start s
1560: 4A        >947           lsr
1561: 4A        >948           lsr
1562: 4A        >949           lsr
1563: D0 02     >950           bne   :ret
1565: A9 0A     >951           lda   #10        ; "0" ==> 10
1567: 60        >952  :ret     rts              ; A = start digit (s)
                >953
                >954  **********************************************************
                >955  *                                                        *
                >956  *                   Clear Register                       *
                >957  *                                                        *
                >958  * At entry: X = Register address                         *
                >959  * At exit:  A = 0, X = $FF                               *
                >960  *                                                        *
                >961  **********************************************************
                >962
1568: 8E 70 15  >963  clear    stx   :clrloop+1 ; Save reg address
156B: A2 05     >964           ldx   #5
156D: A9 00     >965           lda   #0
156F: 95 00     >966  :clrloop sta   0*0,x      ; Clear the register.
1571: CA        >967           dex
1572: 10 FB     >968           bpl   :clrloop
1574: 60        >969           rts
                >970
                >971  **********************************************************
                >972  *                                                        *
                >973  *           Extract NN from 3:2 field of rC              *
                >974  *                                                        *
                >975  * Returns: NN in BCD in 'NN' and Y, in binary in A,      *
                >976  *          X unchanged.                                  *
                >977  *                                                        *
                >978  **********************************************************
                >979
1575: A5 99     >980  midNN    lda   rC+sL      ; Extract NN from xN Nx.
1577: 0A        >981           asl              ; Return binary NN in A.
1578: 0A        >982           asl
1579: 0A        >983           asl
157A: 0A        >984           asl
157B: 85 D1     >985           sta   NN         ; N0
157D: A5 9A     >986           lda   rC+VV      ; Nx (low digit)
157F: 4A        >987           lsr
1580: 4A        >988           lsr
1581: 4A        >989           lsr
1582: 4A        >990           lsr              ; 0N
1583: 05 D1     >991           ora   NN
1585: 85 D1     >992           sta   NN         ; 'NN' = BCD NN
1587: A8        >993           tay
1588: B9 8C 15  >994           lda   bcd2bin,y  ; A = binary NN.
158B: 60        >995           rts
                >996
                >997  * Map 2-digit BCD 00..99 ==> Binary 100..99
                >998
158C: 64 01 02  >999  bcd2bin  db    100,1,2,3,4,5,6,7,8,9 ; BCD 00 ==> 100.
1596: 00 00 00  >1000          ds    6
```

```
159C: 0A 0B 0C >1001          db    10,11,12,13,14,15,16,17,18,19
15A6: 00 00 00 >1002          ds    6
15AC: 14 15 16 >1003          db    20,21,22,23,24,25,26,27,28,29
15B6: 00 00 00 >1004          ds    6
15BC: 1E 1F 20 >1005          db    30,31,32,33,34,35,36,37,38,39
15C6: 00 00 00 >1006          ds    6
15CC: 28 29 2A >1007          db    40,41,42,43,44,45,46,47,48,49
15D6: 00 00 00 >1008          ds    6
15DC: 32 33 34 >1009          db    50,51,52,53,54,55,56,57,58,59
15E6: 00 00 00 >1010          ds    6
15EC: 3C 3D 3E >1011          db    60,61,62,63,64,65,66,67,68,69
15F6: 00 00 00 >1012          ds    6
15FC: 46 47 48 >1013          db    70,71,72,73,74,75,76,77,78,79
1606: 00 00 00 >1014          ds    6
160C: 50 51 52 >1015          db    80,81,82,83,84,85,86,87,88,89
1616: 00 00 00 >1016          ds    6
161C: 5A 5B 5C >1017          db    90,91,92,93,94,95,96,97,98,99
               >1018
               >1019 * $00..$89 B220 character code to ASCII
               >1020
               >1021 b220asc  equ   *          ; B220 code to ASCII
1626: A0       >1022          db    $A0        ; $00 = Blank
1627: 00       >1023          ds    1          ; $01 skip
1628: 00       >1024          db    $00        ; $02 = Ignore
1629: AE A9    >1025          asc   ".)"       ; $03..$04
162B: 00 00 00 >1026          ds    11         ; $05..$0F skip
1636: A8       >1027          asc   "("        ; $10
1637: 00 00    >1028          ds    2          ; $11..$12 skip
1639: AB AA    >1029          asc   "+*"       ; $13..$14
163B: 8C       >1030          db    $8C        ; $15 = Eject
163C: 8D       >1031          db    $8D        ; $16 = CR
163D: 00 00 00 >1032          ds    3+6        ; $17..$1F skip
1646: AD AF    >1033          asc   "-/"       ; $20..$21
1648: 00       >1034          ds    1          ; $22 skip
1649: AC       >1035          asc   ","        ; $23
164A: A5       >1036          asc   "%"        ; $24 (For SNAP CR translation)
164B: 00       >1037          ds    1          ; $25 skip
164C: 89       >1038          db    $89        ; $26 = TAB
164D: A4       >1039          asc   "$"        ; $27
164E: 00 00 00 >1040          ds    2+6+2      ; $28..$31 skip
1658: BF BD A7 >1041          asc   "?='"      ; $32..$34
165B: 00 00 00 >1042          ds    5+6+1      ; $35..$40 skip
1667: C1 C2 C3 >1043          asc   "ABCDEFGHI" ; $41..$49
1670: 00 00 00 >1044          ds    6+1        ; $4A..$50 skip
1677: CA CB CC >1045          asc   "JKLMNOPQR" ; $51..$59
1680: 00 00 00 >1046          ds    6+2        ; $5A..$61 skip
1688: D3 D4 D5 >1047          asc   "STUVWXYZ" ; $62..$69
1690: 00 00 00 >1048          ds    6+16       ; $6A..$7F skip
16A6: B0 B1 B2 >1049          asc   "0123456789" ; $80..$89
```

```
                              ===== Page 85 =====

                 75            put   B220MT
                >1    ***********************************************************
                >2    *                                                         *
                >3    *                Mag Tape Instructions                    *
                >4    *                                                         *
                >5    ***********************************************************
                >6
                >7    blkcnt   equ   line2      ; Block count
                >8    MxRflg   equ   line2+1    ; Flag for MxR op
                >9    compsL   equ   line4      ; sL for compare
                >10   compwd   equ   line4+1    ; Number of comparison word.
                >11   ctlblk   equ   line4+1    ; 'Found ctl block' flag
                >12   ltflag   equ   line8      ; Search found < block.
                >13   mtcptr   equ   line8      ; ptr to preface of mtc block
                >14   keyflg   equ   line8      ; >0 ==> processing key word
                >15   wrdcnt   equ   line8+1    ; Binary word count.
                >16   ctlflg   equ   linev+1    ; Read ctl blocks as normal
                >17
16B0: 88        >18   MTS      dey              ; Y = $FF.
16B1: 84 D5     >19            sty   ctlflg     ; Set 'stop on EOT block' flag.
16B3: A2 04     >20            ldx   #MTUclass  ; Mag Tape class
16B5: 20 58 08 >21             jsr   M_iosel    ; Select device.
16B8: 20 88 08 >22             jsr   M_setlan   ; Set tape lane (0/1).
16BB: A5 9A     >23            lda   rC+VV      ; Decode variant digit.
16BD: 29 04     >24            and   #$04
16BF: D0 66     >25            bne   :done      ; MLS = 4,5,6,7.
16C1: A5 9A     >26            lda   rC+VV
16C3: 29 08     >27            and   #$08
16C5: F0 06     >28            beq   :mtsmfs    ; MTS/MFS = 0,1,2,3
16C7: 20 94 08 >29             jsr   M_resetd   ; MRW/MDA = 8,9.
16CA: 4C 27 17 >30             jmp   :done
                >31
16CD: 85 DC     >32   :mtsmfs  sta   ltflag     ; Clear '<' flag.
16CF: A5 98     >33            lda   rC+S       ; MTS or MFS?
16D1: 29 04     >34            and   #$04
16D3: F0 02     >35            beq   :setsL     ; MTS "field" = 00
16D5: A5 94     >36            lda   rB         ; MFS field = rB:82
16D7: 85 DA     >37   :setsL   sta   compsL     ; Save sL for compare.
16D9: 20 70 08 >38    :nxblk   jsr   M_getwrd   ; Read next word.
16DC: A5 AA     >39            lda   rD+S       ; Isolate sign flag.
16DE: 29 F0     >40            and   #$F0
16E0: C9 B0     >41            cmp   #PREF      ; Block preface word?
16E2: D0 49     >42            bne   ]IOerr3    ; -No, I/O error.
16E4: A5 AB     >43            lda   rD+sL      ; -Yes, save preface
16E6: 85 DD     >44            sta   wrdcnt     ;   word count.
16E8: 20 70 08 >45             jsr   M_getwrd   ; rD = block key word.
16EB: A5 DD     >46            lda   wrdcnt     ; Recover word count.
16ED: 25 D5     >47            and   ctlflg     ; Mask with 'stop on  EOT'.
16EF: C9 01     >48            cmp   #1         ; Is it an EOT block?
16F1: F0 29     >49            beq   :finish    ; -Yes, finish.
16F3: A5 DA     >50            lda   compsL     ; -No, MFS field = rB:82
16F5: 85 99     >51            sta   rC+sL      ;  and fake it in rC.
16F7: A2 AA     >52            ldx   #rD        ; Compare rD w/ search key.
16F9: A0 01     >53            ldy   #1         ; Partial field
16FB: A9 B0     >54            lda   #BCSop     ; Unsigned compare.
16FD: 20 40 0E >55             jsr   compare    ; Do the compare.
1700: A8        >56            tay              ; A state (1,0,-1) to flags.
1701: F0 19     >57            beq   :finish    ; Comparand = key.
1703: 10 0B     >58            bpl   :grtr      ; Comparand > key.
1705: 85 DC     >59            sta   ltflag     ; Comparand < key
1707: 20 A0 08 >60             jsr   M_nxtblk   ; Advance to next block
170A: 88        >61            dey              ; Y = $FF.
170B: 84 D5     >62            sty   ctlflg     ; $FF = 'stop on EOT block'.
170D: 4C D9 16 >63             jmp   :nxblk     ;  and continue search.
                >64
1710: A5 DC     >65   :grtr    lda   ltflag     ; Have we seen < block?
1712: D0 08     >66            bne   :finish    ; -Yes, this is the hit.
```

```
1714: 20 AC 08 >67              jsr   M_prvblk   ; -No, back up 1 block
1717: 84 D5    >68              sty   ctlflg     ; 0 = 'no stop on EOT block'.
1719: 4C D9 16 >69              jmp   :nxblk     ;  and continue search.
               >70
171C: 38       >71    :finish  sec               ; Back ptr up 2 words
171D: A5 CC    >72              lda   ptr        ;  to preface of current block.
171F: E9 0C    >73              sbc   #6*2
1721: 85 CC    >74              sta   ptr
1723: B0 02    >75              bcs   :done      ; No borrow.
1725: C6 CD    >76              dec   ptr+1
1727: 20 64 08 >77    :done    jsr   M_iodsel   ; De-select device.
172A: 4C 01 09 >78              jmp   fetch
               >79
172D: 4C D3 09 >80    ]IOerr3  jmp   IOerr
               >81
1730: A5 9A    >82    MTC      lda   rC+VV      ; Isolate word count.
1732: 29 0F    >83              and   #$0F
1734: D0 02    >84              bne   :nonzero   ; Word count of zero
1736: A9 0A    >85              lda   #10        ;  means tenth word.
1738: 85 DB    >86    :nonzero sta   compwd     ; Save word count.
173A: A5 98    >87              lda   rC+S       ; MTC or MFC?
173C: 29 04    >88              and   #$04
173E: F0 02    >89              beq   :setsL     ; MTC "field" = 00
1740: A5 94    >90              lda   rB         ; MFC field = rB:82
1742: 85 DA    >91    :setsL   sta   compsL     ; Save sL for compare.
1744: A2 04    >92              ldx   #MTUclass  ; Mag Tape class
1746: 20 58 08 >93              jsr   M_iosel    ; Select device.
1749: 20 88 08 >94              jsr   M_setlan   ; Set tape lane (0/1).
174C: A5 CC    >95    :nxblk   lda   ptr        ; Save ptr to preface.
174E: 85 DC    >96              sta   mtcptr
1750: A5 CD    >97              lda   ptr+1
1752: 85 DD    >98              sta   mtcptr+1
1754: 20 70 08 >99              jsr   M_getwrd   ; Read preface word.
1757: A5 AA    >100             lda   rD+S       ; Isolate sign flag.
1759: 29 F0    >101             and   #$F0
175B: C9 B0    >102             cmp   #PREF      ; Block preface word?
175D: D0 CE    >103             bne   ]IOerr3    ; -No, I/O error.
175F: A5 AB    >104             lda   rD+sL      ; Get block word count.
1761: C9 01    >105             cmp   #1         ; Is it an EOT block?
1763: F0 39    >106             beq   :finish    ; -Yes, finish.
1765: A0 00    >107             ldy   #0         ; -No.
1767: B1 CC    >108             lda   (ptr),y    ; Get next word's sign.
1769: C9 07    >109             cmp   #07        ; Is this a control block?
176B: F0 31    >110             beq   :finish    ; -Yes, regard as hit.
176D: C6 DB    >111   :complp  dec   compwd     ; -No. Is comparand next word?
176F: F0 0D    >112             beq   :comp      ; -Yes, compare.
1771: 18       >113   :wrdlp   clc               ; -No, inc ptr to next word.
1772: A5 CC    >114             lda   ptr
1774: 69 06    >115             adc   #6
1776: 85 CC    >116             sta   ptr
1778: 90 F3    >117             bcc   :complp
177A: E6 CD    >118             inc   ptr+1
177C: D0 EF    >119             bne   :complp    ; (always)
               >120
177E: 20 70 08 >121   :comp    jsr   M_getwrd   ; rD = comparand.
1781: A5 DC    >122             lda   mtcptr     ; Restore ptr to
1783: 85 CC    >123             sta   ptr        ;  block preface.
1785: A5 DD    >124             lda   mtcptr+1
1787: 85 CD    >125             sta   ptr+1
1789: A5 DA    >126             lda   compsL     ; Get saved sL
178B: 85 99    >127             sta   rC+sL      ;  and fake it in rC.
178D: A2 AA    >128             ldx   #rD        ; Compare rD w/ scan key.
178F: A0 01    >129             ldy   #1         ; Partial field
1791: A9 B0    >130             lda   #BCSop     ; Unsigned compare.
1793: 20 40 0E >131             jsr   compare    ; Do the compare.
1796: F0 0E    >132             beq   :done      ; -Block key = scan key.
1798: 20 A0 08 >133             jsr   M_nxtblk   ; -Unequal, Adv to nxt block.
```

```
179B: 4C 4C 17 >134           jmp   :nxblk     ;  and continue scan.
               >135
179E: A5 DC    >136  :finish  lda   mtcptr     ; Restore ptr to
17A0: 85 CC    >137           sta   ptr        ;  ctl block preface.
17A2: A5 DD    >138           lda   mtcptr+1
17A4: 85 CD    >139           sta   ptr+1
17A6: 20 64 08 >140  :done    jsr   M_iodsel   ; Deselect device.
17A9: 4C 01 09 >141           jmp   fetch
               >142
17AC: C8       >143  MRR      iny              ; Set MRR flag.
17AD: 84 D9    >144  MRD      sty   MxRflg     ; 1 = MRR, 0 = MRD.
17AF: A5 9A    >145           lda   rC+VV      ; Check variant digit.
17B1: 29 08    >146           and   #$08       ; Isolate and save
17B3: 85 D4    >147           sta   Bmodflg    ;  B-modificatiion flag.
17B5: A5 9A    >148           lda   rC+VV
17B7: 29 01    >149           and   #$01       ; Isolate and save
17B9: 85 D5    >150           sta   ctlflg     ;  ctl blocks normal flag.
17BB: A5 99    >151           lda   rC+sL
17BD: 29 0F    >152           and   #$0F       ; Isolate and save
17BF: D0 02    >153           bne   :stblkct   ;  block count.
17C1: A9 0A    >154           lda   #10        ; Count = 0 ==> 10.
17C3: 85 D8    >155  :stblkct sta   blkcnt
17C5: A2 04    >156           ldx   #MTUclass  ; Mag Tape class.
17C7: 20 58 08 >157           jsr   M_iosel    ; Select device.
17CA: 20 70 08 >158  :blklp   jsr   M_getwrd   ; Preface word to rD.
17CD: A5 AA    >159           lda   rD+S       ; Preface sign byte.
17CF: 29 F0    >160           and   #$F0
17D1: C9 B0    >161           cmp   #PREF      ; Is it flagged as preface?
17D3: D0 64    >162           bne   :ioerr     ; -No, error!
17D5: A9 00    >163           lda   #0         ; -Yes, proceed.
17D7: 85 DB    >164           sta   ctlblk     ; Clear 'found ctl block'
17D9: A4 AB    >165           ldy   rD+sL      ; Block word count (BCD)
17DB: 84 D1    >166           sty   NN         ; Save it.
17DD: B9 8C 15 >167           lda   bcd2bin,y  ; Convert it to binary
17E0: 85 DD    >168           sta   wrdcnt     ;  and save it.
17E2: 85 DC    >169           sta   keyflg     ; First data word is key word.
17E4: A5 D9    >170           lda   MxRflg     ; MRR?
17E6: F0 09    >171           beq   :ckeot     ; -No, don't store preface.
17E8: A5 AA    >172           lda   rD+S       ; -Yes, clear the PREF flag
17EA: 29 0F    >173           and   #$0F       ;  before storing.
17EC: 85 AA    >174           sta   rD+S
17EE: 20 89 18 >175           jsr   strDinc    ; Store preface word for MRR.
17F1: A5 DD    >176  :ckeot   lda   wrdcnt     ; Length = 1 ==> EOT.
17F3: C9 01    >177           cmp   #1         ; End-Of-Tape block?
17F5: F0 45    >178           beq   :eot       ; -Yes, handle it.
17F7: 20 70 08 >179  :wrdlp   jsr   M_getwrd   ; Get next data word.
17FA: A5 AA    >180           lda   rD+S       ; Should this word
17FC: 25 D4    >181           and   Bmodflg    ;  be B-modified?
17FE: F0 03    >182           beq   :noBmod    ; -No.
1800: 20 12 0B >183           jsr   BmodrD     ; -Yes, modify it.
1803: A5 D5    >184  :noBmod  lda   ctlflg     ; Read ctl blocks?
1805: D0 16    >185           bne   :store     ; -Yes, store it.
1807: A5 DB    >186           lda   ctlblk     ; -No. Are we in
1809: C9 07    >187           cmp   #$07       ;   a control block?
180B: D0 06    >188           bne   :notctl    ; -No, continue.
180D: A5 DD    >189           lda   wrdcnt     ; -Yes. Is this the final
180F: C9 01    >190           cmp   #1         ;   (control) word)?
1811: F0 2C    >191           beq   :ctlblk    ; -Yes, handle it.
1813: A5 DC    >192  :notctl  lda   keyflg     ; -No, is this the key word?
1815: F0 06    >193           beq   :store     ; -No, store it.
1817: A5 AA    >194           lda   rD+S       ; -Yes, is this
1819: 29 0F    >195           and   #$0F       ;   a control block?
181B: 85 DB    >196           sta   ctlblk     ; Sign = 7 if control block.
181D: 20 89 18 >197  :store   jsr   strDinc    ; -No, store rD and advance.
1820: A9 00    >198           lda   #0         ; Reset key word
1822: 85 DC    >199           sta   keyflg     ;  (1st word) flag.
1824: C6 DD    >200           dec   wrdcnt     ; More words in block?
```

```
1826: D0 CF    >201           bne   :wrdlp     ; -Yes, continue.
1828: A5 D1    >202           lda   NN         ; Full 100-word block?
182A: F0 03    >203           beq   :noskip    ; -Yes, nothing to skip.
182C: 20 A0 08 >204           jsr   M_nxtblk   ; -No, skip remaining words.
182F: C6 D8    >205  :noskip  dec   blkcnt     ; More blocks?
1831: D0 97    >206           bne   :blklp     ; -Yes, read next block.
1833: 20 64 08 >207           jsr   M_iodsel   ; -No, deselect device.
1836: 4C 01 09 >208           jmp   fetch
               >209
1839: 4C D3 09 >210  :ioerr   jmp   IOerr
               >211
183C: 20 70 08 >212  :eot     jsr   M_getwrd   ; rD = EOT control word.
183F: A6 AD    >213  :ctlblk  ldx   rD+OP      ; Process ctl word in rD.
1841: A4 AC    >214           ldy   rD+VV      ; High 2 digits of aaaa
1843: C0 4A    >215           cpy   #$49+1     ; ADDR error?
1845: B0 3C    >216           bcs   :addrerr   ; -Yes, error!
1847: BD C7 19 >217           lda   BCDLadrl,x ; -No, compute 'memptr'
184A: 79 FB 1A >218           adc   BCDHadrl,y
184D: 85 CA    >219           sta   memptr     ; Low byte of mem address.
184F: BD 61 1A >220           lda   BCDLadrh,x
1852: 79 45 1B >221           adc   BCDHadrh,y
1855: B0 2F    >222           bcs   :undiger   ; Carry out ==> undigit(s)
1857: 85 CB    >223           sta   memptr+1   ; High byte of 'memptr'
1859: A0 05    >224           ldy   #ADDR+1    ; (memptr):04 = rP.
185B: A5 97    >225           lda   rP+1
185D: 91 CA    >226           sta   (memptr),y
185F: 88       >227           dey
1860: A5 96    >228           lda   rP
1862: 91 CA    >229           sta   (memptr),y
1864: 88       >230           dey              ; (memptr):64 = rC:04.
1865: A5 9D    >231           lda   rC+ADDR+1
1867: 91 CA    >232           sta   (memptr),y
1869: 88       >233           dey
186A: A5 9C    >234           lda   rC+ADDR
186C: 91 CA    >235           sta   (memptr),y
186E: A5 AE    >236           lda   rD+ADDR    ; Put bbbb into rP.
1870: 85 96    >237           sta   rP
1872: A5 AF    >238           lda   rD+ADDR+1
1874: 85 97    >239           sta   rP+1
1876: A5 D1    >240           lda   NN         ; Full 100-word block?
1878: F0 03    >241           beq   :nskip     ; -Yes, nothing to skip.
187A: 20 A0 08 >242           jsr   M_nxtblk   ; -No, skip remaining words.
187D: 20 64 08 >243  :nskip   jsr   M_iodsel   ; Deselect device
1880: 4C E3 08 >244           jmp   newP       ;  and branch to bbbb.
               >245
1883: 4C CF 09 >246  :addrerr jmp   ADDRerr
1886: 4C D9 09 >247  :undiger jmp   UNDIGerr
               >248
1889: 20 28 0B >249  strDinc  jsr   storerD    ; (memptr) = rD, inc memptr.
188C: F8       >250           sed              ; / Increment rC:04 (BCD).
188D: 18       >251           clc
188E: A5 9D    >252           lda   rC+ADDR+1
1890: 69 01    >253           adc   #1
1892: 85 9D    >254           sta   rC+ADDR+1
1894: A5 9C    >255           lda   rC+ADDR
1896: 69 00    >256           adc   #0
1898: 85 9C    >257           sta   rC+ADDR
189A: D8       >258           cld              ; \
189B: 60       >259           rts
               >260
189C: C8       >261  MIR      iny              ;
189D: 84 D9    >262  MIW      sty   MxRflg     ; 1 = MIR, 0 = MIW.
189F: A5 99    >263           lda   rC+sL
18A1: 29 0F    >264           and   #$0F       ; Isolate the
18A3: D0 02    >265           bne   :stblkct   ;  block count.
18A5: A9 0A    >266           lda   #10        ; Count = 0 ==> 10.
18A7: 85 D8    >267  :stblkct sta   blkcnt     ; Save block count.
```

```
18A9: A5 9A    >268            lda   rC+VV      ; Word count (BCD)
18AB: 85 D1    >269            sta   NN         ; Save word count.
18AD: A2 04    >270            ldx   #MTUclass  ; Mag Tape class
18AF: 20 58 08 >271            jsr   M_iosel    ; Select device.
18B2: C9 EF    >272            cmp   #EOF       ; -No, are we at EOF?
18B4: D0 27    >273            bne   :ioerr     ; -No, I/O error!
18B6: A5 D9    >274            lda   MxRflg     ; -Yes, MIR or MIW?
18B8: D0 26    >275            bne   :mir       ; -MIR, skip making preface.
18BA: A2 B0    >276            ldx   #rD10      ; -MIW, build preface
18BC: 20 68 15 >277            jsr   clear      ;   word in rD10.
18BF: A5 D1    >278            lda   NN         ; Set word count
18C1: 85 B1    >279            sta   rD10+sL    ;  in 22 field
18C3: A9 B0    >280            lda   #PREF      ;   and preface flag
18C5: 85 B0    >281            sta   rD10+S     ;    in sign.
18C7: A5 D9    >282  :blklp    lda   MxRflg     ; MIR or MIW?
18C9: D0 15    >283            bne   :mir       ; -MIR.
18CB: A2 05    >284            ldx   #5         ; -MIW, copy rD10 to rD.
18CD: B5 B0    >285  :copylp   lda   rD10,x
18CF: 95 AA    >286            sta   rD,x
18D1: CA       >287            dex
18D2: 10 F9    >288            bpl   :copylp
18D4: A6 D1    >289            ldx   NN         ; Restore MIW
18D6: BD 8C 15 >290            lda   bcd2bin,x  ;  binary
18D9: 85 DD    >291            sta   wrdcnt     ;   word count.
18DB: D0 13    >292            bne   :putpref   ; (always)
               >293
18DD: 4C D3 09 >294  :ioerr    jmp   IOerr
               >295
18E0: 20 61 0B >296  :mir      jsr   loadrD     ; Load preface from mem.
18E3: A5 AA    >297            lda   rD+S       ;  Set 'preface' flag
18E5: 09 B0    >298            ora   #PREF      ;   in sign byte,
18E7: 85 AA    >299            sta   rD+S
18E9: A6 AB    >300            ldx   rD+sL      ;    get the word count,
18EB: BD 8C 15 >301            lda   bcd2bin,x  ;     convert to binary,
18EE: 85 DD    >302            sta   wrdcnt     ;      and save it.
18F0: 20 7C 08 >303  :putpref  jsr   M_putwrd   ; Put preface word.
18F3: 20 61 0B >304  :wrdlp    jsr   loadrD     ; Data word to rD
18F6: 20 7C 08 >305            jsr   M_putwrd   ;  and put it.
18F9: C6 DD    >306            dec   wrdcnt     ; More words in block?
18FB: D0 F6    >307            bne   :wrdlp     ; -Yes, continue.
18FD: A5 D1    >308            lda   NN         ; Full 100-word block?
18FF: F0 03    >309            beq   :nskip     ; -Yes, nothing to skip.
1901: 20 A0 08 >310            jsr   M_nxtblk   ; -No, skip remaining words.
1904: C6 D8    >311  :nskip    dec   blkcnt     ; More blocks?
1906: D0 BF    >312            bne   :blklp     ; -Yes, continue.
1908: A9 EF    >313            lda   #EOF       ; -No, set EOF.
190A: A0 00    >314            ldy   #0
190C: 8D 04 C0 >315            sta   WRITMAIN
190F: 91 CC    >316            sta   (ptr),y
1911: 8D 05 C0 >317            sta   WRITAUX
1914: 20 64 08 >318            jsr   M_iodsel   ; Deselect device.
1917: 4C 01 09 >319            jmp   fetch
               >320
191A: C8       >321  MOR       iny
191B: 84 D9    >322  MOW       sty   MxRflg     ; 1 = MOR, 0 = MOW.
191D: A5 99    >323            lda   rC+sL
191F: 29 0F    >324            and   #$0F       ; Isolate the
1921: D0 02    >325            bne   :stblkct   ;  block count.
1923: A9 0A    >326            lda   #10        ; Count = 0 ==> 10.
1925: 85 D8    >327  :stblkct  sta   blkcnt     ; Save block count.
1927: A5 9A    >328            lda   rC+VV      ; MOW word count (BCD)
1929: 85 D1    >329            sta   NN         ; Save MOW word count.
192B: A2 04    >330            ldx   #MTUclass  ; Mag Tape class
192D: 20 58 08 >331            jsr   M_iosel    ; Select device.
1930: C9 EF    >332  :blklp    cmp   #EOF       ; Are we at end-of-file?
1932: F0 3F    >333            beq   :ioerr     ; -Yes, I/O error!
1934: 20 70 08 >334            jsr   M_getwrd   ; -No, read preface.
```

```
1937: A5 AA    >335           lda   rD+S        ; Preface flag/sign byte
1939: 29 F0    >336           and   #$F0        ; Isolate flag.
193B: C9 B0    >337           cmp   #PREF       ; Is this a preface?
193D: D0 34    >338           bne   :ioerr      ; -No, block sync error!
193F: A5 D9    >339           lda   MxRflg      ; -Yes. MOR or MOW?
1941: F0 09    >340           beq   :mow        ; -MOW (use OP's NN)
1943: A0 01    >341           ldy   #sL         ; -MOR, compare with
1945: B1 CA    >342           lda   (memptr),y ;   memory preface.
1947: 85 D1    >343           sta   NN          ; Save in NN.
1949: 20 CB 08 >344           jsr   incmem      ; Advance memptr to data.
194C: A5 D1    >345  :mow     lda   NN          ; Compare NN
194E: C5 AB    >346           cmp   rD+sL       ;  with preface.
1950: D0 21    >347           bne   :ioerr      ; Preface mismatch!
1952: A8       >348           tay
1953: B9 8C 15 >349           lda   bcd2bin,y  ; Convert NN to
1956: 85 DD    >350           sta   wrdcnt      ;  binary word count.
1958: 20 61 0B >351  :wrdlp   jsr   loadrD      ; Data word to rD
195B: 20 7C 08 >352           jsr   M_putwrd    ;  and put it to file.
195E: C6 DD    >353           dec   wrdcnt      ; More words in block?
1960: D0 F6    >354           bne   :wrdlp      ; -Yes, continue.
1962: A5 D1    >355           lda   NN          ; Full 100-word block?
1964: F0 03    >356           beq   :noskip     ; -Yes, don't skip rest.
1966: 20 A0 08 >357           jsr   M_nxtblk    ; -No, skip to next block.
1969: C6 D8    >358  :noskip  dec   blkcnt      ; More blocks?
196B: D0 C3    >359           bne   :blklp      ; -Yes, continue.
196D: 20 64 08 >360           jsr   M_iodsel    ; Deselect device.
1970: 4C 01 09 >361           jmp   fetch
               >362
1973: 4C D3 09 >363  :ioerr   jmp   IOerr
               >364
1976: A5 99    >365  MPF      lda   rC+sL       ; Get block count.
1978: 29 0F    >366           and   #$0F
197A: D0 02    >367           bne   :setblk
197C: A9 0A    >368           lda   #10         ; '0' ==> 10.
197E: 85 D8    >369  :setblk  sta   blkcnt      ; Save block count.
1980: A2 04    >370           ldx   #MTUclass  ; Mag Tape class
1982: 20 58 08 >371           jsr   M_iosel     ; Select the device.
1985: A8       >372           tay               ; Save next flag byte.
1986: A5 9A    >373           lda   rC+VV       ; MPF, MPB, oe MPE?
1988: 29 0F    >374           and   #$0F        ; Isolate variant digit.
198A: C9 01    >375           cmp   #1
198C: F0 11    >376           beq   :mpb        ; Mag tape Position Backward.
198E: C9 02    >377           cmp   #2
1990: F0 16    >378           beq   :mpe        ; Mag tape Position at End.
1992: 20 A0 08 >379  :mpf     jsr   M_nxtblk    ; MPF, advance to next block.
1995: C6 D8    >380           dec   blkcnt      ; More blocks to skip?
1997: D0 F9    >381           bne   :mpf        ; -Yes, keep going.
1999: 20 64 08 >382  :done    jsr   M_iodsel    ; -No, deselect the device.
199C: 4C 01 09 >383           jmp   fetch
               >384
199F: 20 AC 08 >385  :mpb     jsr   M_prvblk    ; Position to previous block.
19A2: C6 D8    >386           dec   blkcnt      ; More blocks to skip?
19A4: D0 F9    >387           bne   :mpb        ; -Yes, continue.
19A6: F0 F1    >388           beq   :done       ; -No, done. (always)
               >389
19A8: 98       >390  :mpe     tya               ; Recover next flag byte.
19A9: C9 EF    >391  :mpelp   cmp   #EOF        ; At End-Of-File?
19AB: F0 EC    >392           beq   :done       ; -Yes, done!
19AD: 20 A0 08 >393           jsr   M_nxtblk    ; -No, adv to next block
19B0: 4C A9 19 >394           jmp   :mpelp      ;       and check for EOF.
               >395
19B3: A5 9A    >396  MIB      lda   rC+VV       ; MIB or MIE
19B5: 29 0F    >397           and   #$0F        ; Isolate variant digit.
19B7: C9 01    >398           cmp   #1          ; Is it MIE?
19B9: D0 03    >399           bne   :mib        ; -No, it's an MIB.
19BB: 4C 01 09 >400  :nop     jmp   fetch       ; -Yes, MIE = NOP.
19BE: A5 99    >401  :mib     lda   rC+sL
```

===== Page 91 =====

```
19C0: 29 E0    >402           and   #$E0       ; Is unit = 0 or 1?
19C2: D0 F7    >403           bne   :nop       ; -No, so it's a NOP.
19C4: 4C EF 12 >404           jmp   BUN        ; -Yes, so it's a BUN.
```

```
                76            put   B220BCDTBL
               >1    * 4-digit BCD to binary word address tables
               >2
               >3    BCDLadrl equ   *          ; BCD lo 2 dig --> addr lo byte
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19C7: D0       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19C8: D6       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19C9: DC       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19CA: E2       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19CB: E8       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19CC: EE       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19CD: F4       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19CE: FA       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19CF: 00       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D0: 06       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D1: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D2: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D3: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D4: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D5: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D6: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D7: 0C       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D8: 12       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19D9: 18       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19DA: 1E       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19DB: 24       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19DC: 2A       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19DD: 30       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19DE: 36       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19DF: 3C       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19E0: 42       >12            db    <]T*10+]U*6+MEM
```

===== Page 94 =====

```
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E1: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E2: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E3: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E4: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E5: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E6: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E7: 48       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E8: 4E       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19E9: 54       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19EA: 5A       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19EB: 60       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19EC: 66       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
19ED: 6C       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19EE: 72       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19EF: 78       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F0: 7E       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F1: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F2: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F3: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F4: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F5: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F6: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F7: 84       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F8: 8A       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19F9: 90       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19FA: 96       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
```

===== Page 96 =====

```
19FB: 9C       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19FC: A2       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19FD: A8       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19FE: AE       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
19FF: B4       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A00: BA       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A01: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A02: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A03: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A04: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A05: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A06: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A07: C0       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A08: C6       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A09: CC       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A0A: D2       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A0B: D8       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A0C: DE       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A0D: E4       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A0E: EA       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A0F: F0       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A10: F6       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A11: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A12: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A13: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A14: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A15: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
```

```
                              ===== Page 98 =====

               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A16: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A17: FC       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A18: 02       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A19: 08       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A1A: 0E       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A1B: 14       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A1C: 1A       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A1D: 20       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A1E: 26       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A1F: 2C       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A20: 32       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A21: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A22: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A23: 00       >10            db    0
```

===== Page 99 =====

```
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A24: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A25: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A26: 00       >10              db    0
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A27: 38       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A28: 3E       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A29: 44       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A2A: 4A       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A2B: 50       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A2C: 56       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A2D: 5C       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A2E: 62       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A2F: 68       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7     ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8     ]U        equ   ]Ax-]A0    ; BCD units digit
1A30: 6E       >12              db    <]T*10+]U*6+MEM
               >5     ]Ax       equ   *-BCDLadrl ; ]Ax = index of table entry
               >6     ]T        equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A31: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A32: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A33: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A34: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A35: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A36: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A37: 74       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A38: 7A       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A39: 80       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A3A: 86       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A3B: 8C       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A3C: 92       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A3D: 98       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
```

```
1A3E: 9E       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A3F: A4       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A40: AA       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A41: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A42: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A43: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A44: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A45: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A46: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A47: B0       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A48: B6       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A49: BC       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A4A: C2       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A4B: C8       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A4C: CE       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A4D: D4       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A4E: DA       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A4F: E0       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A50: E6       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A51: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A52: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A53: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A54: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A55: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A56: 00       >10            db    0
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A57: EC       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A58: F2       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A59: F8       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A5A: FE       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A5B: 04       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A5C: 0A       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A5D: 10       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A5E: 16       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A5F: 1C       >12            db    <]T*10+]U*6+MEM
               >5    ]Ax      equ   *-BCDLadrl ; ]Ax = index of table entry
               >6    ]T       equ   ]Ax/16     ; BCD tens digit
               >7    ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >8    ]U       equ   ]Ax-]A0    ; BCD units digit
1A60: 22       >12            db    <]T*10+]U*6+MEM
               >15
               >16   BCDLadrh equ   *          ; BCD lo 2 dig --> addr hi byte
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A61: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A62: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A63: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A64: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A65: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A66: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A67: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A68: 4A       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A69: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A6A: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A6B: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A6C: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A6D: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A6E: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A6F: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A70: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A71: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A72: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A73: 4B       >25            db    >]T*10+]U*6+MEM
```

```
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A74: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A75: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A76: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A77: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A78: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A79: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A7A: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A7B: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A7C: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A7D: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A7E: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A7F: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A80: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A81: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A82: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A83: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A84: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A85: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A86: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A87: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A88: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A89: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A8A: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A8B: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A8C: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A8D: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
```

```
1A8E: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A8F: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A90: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A91: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A92: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A93: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A94: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A95: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A96: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A97: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A98: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A99: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A9A: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A9B: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A9C: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A9D: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A9E: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1A9F: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA0: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA1: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA2: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA3: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA4: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA5: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA6: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA7: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA8: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
```

```
                              ===== Page 109 =====

               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AA9: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AAA: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AAB: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AAC: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AAD: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AAE: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AAF: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB0: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB1: 4B       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB2: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB3: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB4: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB5: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB6: 4C       >25            db    >]T*10+]U*6+MEM
```

```
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB7: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB8: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AB9: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ABA: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ABB: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ABC: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ABD: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ABE: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ABF: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC0: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC1: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC2: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC3: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC4: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC5: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC6: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC7: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC8: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AC9: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ACA: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ACB: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ACC: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ACD: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ACE: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ACF: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD0: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
```

```
1AD1: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD2: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD3: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD4: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD5: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD6: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD7: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD8: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AD9: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ADA: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ADB: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ADC: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ADD: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1ADE: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1ADF: FF       >23             db    $FF        ; Force overflow on undigits.
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE0: FF       >23             db    $FF        ; Force overflow on undigits.
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE1: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE2: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE3: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE4: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE5: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE6: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE7: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE8: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AE9: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AEA: 4C       >25             db    >]T*10+]U*6+MEM
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U        equ   ]Ax-]A0    ; BCD units digit
1AEB: FF       >23             db    $FF        ; Force overflow on undigits.
               >18   ]Ax       equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T        equ   ]Ax/16     ; BCD tens digit
               >20   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AEC: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AED: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AEE: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AEF: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF0: FF       >23            db    $FF        ; Force overflow on undigits.
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF1: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF2: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF3: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF4: 4C       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF5: 4D       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF6: 4D       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF7: 4D       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF8: 4D       >25            db    >]T*10+]U*6+MEM
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AF9: 4D       >25            db    >]T*10+]U*6+MEM
```

===== Page 115 =====

```
               >18   ]Ax      equ   *-BCDLadrh ; ]Ax = index of table entry
               >19   ]T       equ   ]Ax/16     ; BCD tens digit
               >20   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >21   ]U       equ   ]Ax-]A0    ; BCD units digit
1AFA: 4D       >25            db    >]T*10+]U*6+MEM
               >28
               >29   BCDHadrl equ   *          ; BCD Hi 2 dig --> bin lo byte
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1AFB: 00       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1AFC: 58       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1AFD: B0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1AFE: 08       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1AFF: 60       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B00: B8       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B01: 10       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B02: 68       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B03: C0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B04: 18       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B05: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B06: 00       >36            db    0
```

```
                              ===== Page 116 =====

               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B07: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B08: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B09: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B0A: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B0B: 70       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B0C: C8       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B0D: 20       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B0E: 78       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B0F: D0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B10: 28       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B11: 80       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B12: D8       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B13: 30       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B14: 88       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B15: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B16: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B17: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B18: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B19: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B1A: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B1B: E0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B1C: 38       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B1D: 90       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B1E: E8       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B1F: 40       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B20: 98       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
```

```
1B21: F0       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B22: 48       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B23: A0       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B24: F8       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B25: 00       >36             db    0
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B26: 00       >36             db    0
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B27: 00       >36             db    0
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B28: 00       >36             db    0
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B29: 00       >36             db    0
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B2A: 00       >36             db    0
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B2B: 50       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B2C: A8       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B2D: 00       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T        equ   ]Ax/16     ; BCD tens digit
               >33   ]A0       equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U        equ   ]Ax-]A0    ; BCD units digit
1B2E: 58       >38             db    <]T*10+]U*600
               >31   ]Ax       equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B2F: B0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B30: 08       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B31: 60       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B32: B8       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B33: 10       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B34: 68       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B35: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B36: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B37: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B38: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B39: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B3A: 00       >36            db    0
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B3B: C0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B3C: 18       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B3D: 70       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B3E: C8       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B3F: 20       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B40: 78       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B41: D0       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B42: 28       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B43: 80       >38            db    <]T*10+]U*600
               >31   ]Ax      equ   *-BCDHadrl ; ]Ax = index of table entry
               >32   ]T       equ   ]Ax/16     ; BCD tens digit
               >33   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >34   ]U       equ   ]Ax-]A0    ; BCD units digit
1B44: D8       >38            db    <]T*10+]U*600
               >41
               >42   BCDHadrh equ   *          ; BCD Hi 2 dig --> bin Hi byte
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B45: 00       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B46: 02       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B47: 04       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B48: 07       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B49: 09       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B4A: 0B       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B4B: 0E       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B4C: 10       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B4D: 12       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B4E: 15       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B4F: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B50: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B51: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B52: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B53: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B54: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B55: 17       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B56: 19       >51            db    >]T*10+]U*600
```

```
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B57: 1C       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B58: 1E       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B59: 20       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B5A: 23       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B5B: 25       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B5C: 27       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B5D: 2A       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B5E: 2C       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B5F: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B60: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B61: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B62: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B63: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B64: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B65: 2E       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B66: 31       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B67: 33       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B68: 35       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B69: 38       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B6A: 3A       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B6B: 3C       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B6C: 3F       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B6D: 41       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B6E: 43       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B6F: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B70: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
```

```
1B71: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B72: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B73: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B74: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B75: 46       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B76: 48       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B77: 4B       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B78: 4D       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B79: 4F       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B7A: 52       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B7B: 54       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B7C: 56       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B7D: 59       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B7E: 5B       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B7F: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B80: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B81: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B82: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B83: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B84: FF       >49            db    $FF        ; Force overflow on undigits.
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B85: 5D       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B86: 60       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B87: 62       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B88: 64       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B89: 67       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B8A: 69       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
               >47   ]U       equ   ]Ax-]A0    ; BCD units digit
1B8B: 6B       >51            db    >]T*10+]U*600
               >44   ]Ax      equ   *-BCDHadrh ; ]Ax = index of table entry
               >45   ]T       equ   ]Ax/16     ; BCD tens digit
               >46   ]A0      equ   ]T*16      ; ]A0 = index w/ lo digit = 0
```

```
              >47   ]U       equu   ]Ax-]A0    ; BCD units digit
1B8C: 6E      >51            db     >]T*10+]U*600
              >44   ]Ax      equ    *-BCDHadrh ; ]Ax = index of table entry
              >45   ]T       equ    ]Ax/16     ; BCD tens digit
              >46   ]A0      equ    ]T*16      ; ]A0 = index w/ lo digit = 0
              >47   ]U       equ    ]Ax-]A0    ; BCD units digit
1B8D: 70      >51            db     >]T*10+]U*600
              >44   ]Ax      equ    *-BCDHadrh ; ]Ax = index of table entry
              >45   ]T       equ    ]Ax/16     ; BCD tens digit
              >46   ]A0      equ    ]T*16      ; ]A0 = index w/ lo digit = 0
              >47   ]U       equ    ]Ax-]A0    ; BCD units digit
1B8E: 72      >51            db     >]T*10+]U*600
              >54
              >55   simend   equ    *-1        ; End of B220SIM code
              >56            err    simend/MEM ; Can't encroach on MEM area.
```

```
                            ===== Page 127 =====

                            org              ; Reestablish code offset
                 AUXend     equ   *          ; End of Aux code
                            err   */$9600    ; Total code limit.


--End assembly, 8181 bytes, Errors: 0


Symbol table - alphabetical order:

   ADCYop  =$79        ADCZop  =$65        ADD     =$0C2A      ADDR    =$04
   ADDRerr =$09CF      ADDRerrR=$08D7      ADL     =$0C9A      ALTCHAR =$C00F
   AR1     =$0700      AR2     =$0680      AR4     =$0600      AR8     =$0580
   ARbord  =$0DD4      ARmid   =$0DFA      ARv     =$0428      AUXcode =$153D
   AUXend  =$27F5      AUXrts  =$08C4      Aattr   =$0C9A      Acol    =$05
   Ain     =$0A75      Alab    =$0583      Aparm   =$14E6    ? B220SIM =$0800
   B220col =$0C        B220end =$C7        B220msg =$0DBF      B220strt=$90
   BASCALC =$FBC1      BASL    =$28        BCDHadrh=$1B45      BCDHadrl=$1AFB
   BCDLadrh=$1A61      BCDLadrl=$19C7      BCE     =$12DD      BCH     =$12C9
   BCS     =$1355      BCSop   =$B0        BEEP    =$FBDD      BFA     =$1306
   BFR     =$1302      BITZop  =$24        BNEop   =$D0        BOF     =$12AC
   BPC1    =$0728      BPC2    =$06A8      BPC4    =$0628      BPC8    =$05A8
   BPCbord =$0E20      BPCmid  =$0E46      BPCv    =$0450      BRP     =$12B9
   BSA     =$12BF      BSSTATE =$BE42      BUN     =$12EF      Battr   =$0CCA
   Bcol    =$05        Bin     =$0A79      Blab    =$05AB      Bmodflg =$D4
   BmodrD  =$0B12      Bparm   =$14EE      CAA     =$0C23      CAD     =$0C0A
   CFA     =$0E20      CH      =$24        CLA     =$142D      CLCop   =$18
   CLL     =$144E      CMPIop  =$C9        COMP    =$C2        COMPcol =$19
   COUT    =$FDED      CROUT   =$FD8E      CSU     =$0BF5      CSW     =$B6
   Cattr   =$0CEA      Ccol    =$15        Cin     =$0A7D      Clab    =$05BB
   DBB     =$1299      DFL     =$1164      DIV     =$0D5B      DLB     =$1174
   DOSCMD  =$BE03      DOSCON  =$03D0      EMPTY   =$EE        EOB     =$EB
   EOF     =$EF        ERR     =$C1        ERRcol  =$15        ERRlab  =$0567
   EXP     =$01        EXT     =$0DF8      Eparm   =$14EA      FAD     =$0EDE
   FDV     =$109B      FIELDerr=$09CB      FMU     =$1000      FSU     =$0FEB
   HLT     =$0AA0      HOME    =$FC58      Help1   =$0E93    ? Help2   =$0EB8
?  Help3   =$0EDE    ? Help4   =$0F01      IBB     =$1286      IFL     =$111E
   IN      =$0200      INDshow =$0FCC      IOerr   =$09D3      KAD     =$095C
   KBD     =$C000      KBSTROBE=$C010      LDB     =$13E9      LDR     =$13DD
   LSA     =$140F      MANT    =$02        MEM     =$4AD0      MIB     =$19B3
   MIR     =$189C      MIW     =$189D      MOR     =$191A      MOW     =$191B
   MPF     =$1976      MRD     =$17AD      MRR     =$17AC      MTC     =$1730
   MTS     =$16B0      MTUclass=$04        MUL     =$0CD2      M_COUT  =$08B8
   M_disp  =$084C      M_getwrd=$0870      M_iodsel=$0864      M_iosel =$0858
   M_keyin =$083A      M_nxtblk=$08A0      M_prvblk=$08AC      M_putwrd=$087C
   M_resetd=$0894      M_setlan=$0888      M_stop  =$0843      MxRflg  =$D9
   NN      =$D1        NOP     =$0AA0      NOPop   =$EA        OFLcol  =$1F
   OFLerr  =$09C7      OP      =$03        OPerr   =$09C3      Ov      =$C3
   OvHlt   =$C7        PDebx   =$14AA      PDfae   =$1472      PRB     =$0AA9
   PRBL2   =$F94A      PRD     =$0AA3      PREF    =$B0        PRI     =$0B35
?  PRINTERR=$BE0C      PTPclass=$02        PTRclass=$00        PWI     =$0B6E
   PWR     =$0B38      Pattr   =$0CDA      Pcol    =$0D        Pin     =$0A81
   Plab    =$05B3      READAUX =$C003      READMAIN=$C002    ? RESTART =$0803
   RND     =$0DD6      RPTcol  =$22        RTF     =$1220      RUN     =$C0
   RUNcol  =$11        Rattr   =$0CB2      Rcol    =$17        Rin     =$0A85
   Rlab    =$0595      Rp      =$C4        S       =$00        SBCYop  =$F9
   SBCZop  =$E5        SECop   =$38        SLA     =$148E      SOR     =$1362
   SPKR    =$C030      SPO     =$0B71      SRA     =$1459      STA     =$1376
   STAT    =$0E6C      STATlin =$0550      STP     =$1418      SUB     =$0CBC
   SW1col  =$06        SWlab   =$0553    ? TABV    =$FB5B      UNDIGerR=$08DA
   UNDIGerr=$09D9      VV      =$02        WNDTOP  =$22        WRITAUX =$C005
   WRITMAIN=$C004      X_IOerr =$0821      X_cont  =$0818      X_fetch =$0806
   X_incP  =$082A      X_newP  =$080F    V ]A0     =$40      V ]Ax     =$49
V  ]IOerr1 =$122F    V ]IOerr2 =$142F    V ]IOerr3 =$172D    V? ]Ov     =$1215
V  ]T      =$04      V ]U      =$09      V? ]adc    =$11CC   V ]add    =$0C3C
V? ]bfr    =$1308    V? ]clc    =$11BD   V? ]cmp    =$11CE   V? ]contin =$0950
V? ]dfl    =$1189    V ]err    =$09DB    V ]errpt  =$1171    V ]fad    =$0EEC
```

```
V  ]fetch1 =$142A   V  ]fetch2 =$13C0   V? ]fetch3 =$12FF   V  ]fetch4 =$1161
V  ]incptr6=$120D   V  ]keep   =$15     V  ]kend   =$15     V? ]nop    =$11F6
V? ]prd    =$0ABD   V  ]resptr =$1340   V? ]rts    =$127D   V  ]stop   =$095C
V? ]sub    =$11F9      advoff  =$1328   MD align   =$8000   MD auxjmp  =$8000
MD auxjsr  =$8000      b220asc =$1626      backoff =$1308      bcd2bin =$158C
   beepget =$0AAD      bfdirty =$1071      bfend   =$1068      bffn    =$106C
   bflane  =$1070      bfoff   =$106D      bfptr   =$1066      bfsiz   =$106A
   bfstart =$1064      blanklin=$0D8D      blkcnt  =$D8        blksize =$025E
   bload   =$14D8      bsave   =$14DF      changed =$DA        ckpref  =$12BC
   classdbx=$10B8      clear   =$1568      clearAR =$1090      common  =$0800
   compare =$0E40      compsL  =$DA        compwd  =$DB        ctlblk  =$DB
   ctlflg  =$D5        cursor  =$57        db      =$1064      dbsz    =$0E
   dbx     =$D2        decblk  =$12F6      delete  =$FF        disARmid=$0D95
   disBPCbo=$0DA3      disBPCmi=$0DB1      disiocfg=$0B76      dispA   =$0F45
   dispB   =$0F53      dispC   =$0F61      dispP   =$0F5A      dispR   =$0F4C
   dispSTAT=$0F68      dispcnt =$64        dispctr =$D3        dispdig =$1033
   disphelp=$0F22      display =$0F33      disppanl=$0D04      dispreg =$0FF5
   divide  =$0D61      dnarrow =$8A        doread  =$1432      dowrite =$13F4
   ediocfg =$0B6D      emptydb =$1392      endcomm =$08D7      escape  =$9B
   exchAR  =$1546      execute =$0922      fetch   =$0901      flushall=$13D0
   flushbuf=$13E3      fnamecol=$0C        fnames  =$1100      fnlen   =$19
   fnx     =$D4        fnxdbx  =$10BE      fnxfn   =$10C6      getdig  =$0AB0
   getwrd  =$11F9      incP    =$09A3      incblk  =$12AA      incmem  =$08CB
   init    =$08D7      initstk =$0954      inptr   =$CE        instptr =$C8
   intabl  =$0A75      inverse =$0C8D      iocfgstr=$0AC7      iocfgtt =$0B
   iodsel  =$11EC      iosel   =$11C8      keyflg  =$DC        keyin   =$0955
   keyinR  =$08DD      line    =$D9        line1   =$D6        line2   =$D8
   line4   =$DA        line8   =$DC        linev   =$D4        loadrA  =$0C14
   loadrD  =$0B61      ltarrow =$88        ltflag  =$DC     MD mainjmp =$8000
MD mainjsr =$8000      memb    =$7530      memptr  =$CA        midNN   =$1575
   mt0bf   =$64B4      mt1bf   =$7C62      mtbfsz  =$17AC      mtcptr  =$DC
   multiply=$0CD8      ndb     =$06        newP    =$08E3      newp    =$C5
   noAD    =$8000      nxtblk  =$127E      off     =$A0        on      =$AA
   operr   =$89C3      optabh  =$0A46      optabl  =$09EC   ?  pdoscmd =$1520
   pdosxeq =$1525      prvblk  =$12D2      ptbfsz  =$0258      ptpch0bf=$6000
   ptpch1bf=$625A      ptr     =$CC        ptrdr0bf=$3B4C      ptrdr1bf=$3DA6
   putbyte =$14F6      putpdcmd=$150E      putwdhx =$14F2      putwrd  =$1232
   rA      =$9E        rB      =$94        rBx     =$90        rC      =$98
   rD      =$AA        rD10    =$B0        rP      =$96        rR      =$A4
   readbuf =$1273      reset   =$0937      resetdb =$1384      resetdbs=$1375
MD resi    =$8000      restart =$0947      rtmargin=$04        sL      =$01
   savex   =$D7        selBASL =$DC        selch   =$D8        selected=$D5
   selsave =$D6     MD seti    =$8000      setlan  =$134B      setptr  =$11DB
   shleft1 =$0A89      signtbl =$0E10      simend  =$1B8E      skipincP=$C6
   slA     =$153B      slT     =$1531      splitsL =$1554      srA     =$1502
   srAM    =$1504      srAMR   =$150F      srAS    =$1500   ?  srR     =$1512
   srT     =$150D      srT2    =$151D      stopR   =$08E0      storerD =$0B28
   strDinc =$1889      t1      =$D0        tabs    =$0BF0      uparrow =$8B
   wrdcnt  =$DD        xeqflg  =$D5        zeroff  =$D6


Symbol table - numerical order:

   S       =$00        PTRclass=$00        sL      =$01        EXP     =$01
   VV      =$02        MANT    =$02        PTPclass=$02        OP      =$03
   ADDR    =$04        rtmargin=$04        MTUclass=$04     V  ]T      =$04
   Acol    =$05        Bcol    =$05        ndb     =$06        SW1col  =$06
V  ]U      =$09        iocfgtt =$0B        fnamecol=$0C        B220col =$0C
   Pcol    =$0D        dbsz    =$0E        RUNcol  =$11        Ccol    =$15
   ERRcol  =$15     V  ]keep   =$15     V  ]kend   =$15        Rcol    =$17
   CLCop   =$18        COMPcol =$19        fnlen   =$19        OFLcol  =$1F
   WNDTOP  =$22        RPTcol  =$22        BITZop  =$24        CH      =$24
   BASL    =$28        SECop   =$38     V  ]A0     =$40     V  ]Ax     =$49
   cursor  =$57        dispcnt =$64        ADCZop  =$65        ADCYop  =$79
   ltarrow =$88        dnarrow =$8A        uparrow =$8B        B220strt=$90
   rBx     =$90        rB      =$94        rP      =$96        rC      =$98
   escape  =$9B        rA      =$9E        off     =$A0        rR      =$A4
   rD      =$AA        on      =$AA        BCSop   =$B0        PREF    =$B0
   rD10    =$B0        CSW     =$B6        RUN     =$C0        ERR     =$C1
   COMP    =$C2        Ov      =$C3        Rp      =$C4        newp    =$C5
   skipincP=$C6        B220end =$C7        OvHlt   =$C7        instptr =$C8
   CMPIop  =$C9        memptr  =$CA        ptr     =$CC        inptr   =$CE
   BNEop   =$D0        t1      =$D0        NN      =$D1        dbx     =$D2
   dispctr =$D3        linev   =$D4        fnx     =$D4        Bmodflg =$D4
   selected=$D5        xeqflg  =$D5        ctlflg  =$D5        line1   =$D6
   selsave =$D6        zeroff  =$D6        savex   =$D7        line2   =$D8
   selch   =$D8        blkcnt  =$D8        line    =$D9        MxRflg  =$D9
   line4   =$DA        changed =$DA        compsL  =$DA        compwd  =$DB
   ctlblk  =$DB        line8   =$DC        selBASL =$DC        ltflag  =$DC
   mtcptr  =$DC        keyflg  =$DC        wrdcnt  =$DD        SBCZop  =$E5
   NOPop   =$EA        EOB     =$EB        EMPTY   =$EE        EOF     =$EF
   SBCYop  =$F9        delete  =$FF        IN      =$0200      ptbfsz  =$0258
   blksize =$025E      DOSCON  =$03D0      ARv     =$0428      BPCv    =$0450
   STATlin =$0550      SWlab   =$0553      ERRlab  =$0567      AR8     =$0580
   Alab    =$0583      Rlab    =$0595      BPC8    =$05A8      Blab    =$05AB
   Plab    =$05B3      Clab    =$05BB      AR4     =$0600      BPC4    =$0628
   AR2     =$0680      BPC2    =$06A8      AR1     =$0700      BPC1    =$0728
   common  =$0800   ?  B220SIM =$0800   ?  RESTART =$0803      X_fetch =$0806
   X_newP  =$080F      X_cont  =$0818      X_IOerr =$0821      X_incP  =$082A
   M_keyin =$083A      M_stop  =$0843      M_disp  =$084C      M_iosel =$0858
   M_iodsel=$0864      M_getwrd=$0870      M_putwrd=$087C      M_setlan=$0888
   M_resetd=$0894      M_nxtblk=$08A0      M_prvblk=$08AC      M_COUT  =$08B8
   AUXrts  =$08C4      incmem  =$08CB      endcomm =$08D7      init    =$08D7
   ADDRerrR=$08D7      UNDIGerR=$08DA      keyinR  =$08DD      stopR   =$08E0
   newP    =$08E3      fetch   =$0901      execute =$0922      reset   =$0937
   restart =$0947   V? ]contin =$0950      initstk =$0954      keyin   =$0955
V  ]stop   =$095C      KAD     =$095C      incP    =$09A3      OPerr   =$09C3
   OFLerr  =$09C7      FIELDerr=$09CB      ADDRerr =$09CF      IOerr   =$09D3
   UNDIGerr=$09D9   V  ]err    =$09DB      optabl  =$09EC      optabh  =$0A46
   intabl  =$0A75      Ain     =$0A75      Bin     =$0A79      Cin     =$0A7D
   Pin     =$0A81      Rin     =$0A85      shleft1 =$0A89      HLT     =$0AA0
   NOP     =$0AA0      PRD     =$0AA3      PRB     =$0AA9      beepget =$0AAD
   getdig  =$0AB0   V? ]prd    =$0ABD      iocfgstr=$0AC7      BmodrD  =$0B12
   storerD =$0B28      PRI     =$0B35      PWR     =$0B38      loadrD  =$0B61
   ediocfg =$0B6D      PWI     =$0B6E      SPO     =$0B71      disiocfg=$0B76
   tabs    =$0BF0      CSU     =$0BF5      CAD     =$0C0A      loadrA  =$0C14
   CAA     =$0C23      ADD     =$0C2A   V  ]add    =$0C3C      inverse =$0C8D
   Aattr   =$0C9A      ADL     =$0C9A      Rattr   =$0CB2      SUB     =$0CBC
   Battr   =$0CCA      MUL     =$0CD2      multiply=$0CD8      Pattr   =$0CDA
   Cattr   =$0CEA      disppanl=$0D04      DIV     =$0D5B      divide  =$0D61
   blanklin=$0D8D      disARmid=$0D95      disBPCbo=$0DA3      disBPCmi=$0DB1
   B220msg =$0DBF      ARbord  =$0DD4      RND     =$0DD6      EXT     =$0DF8
   ARmid   =$0DFA      signtbl =$0E10      BPCbord =$0E20      CFA     =$0E20
   compare =$0E40      BPCmid  =$0E46      STAT    =$0E6C      Help1   =$0E93
?  Help2   =$0EB8   ?  Help3   =$0EDE      FAD     =$0EDE   V  ]fad    =$0EEC
?  Help4   =$0F01      disphelp=$0F22      display =$0F33      dispA   =$0F45
   dispR   =$0F4C      dispB   =$0F53      dispP   =$0F5A      dispC   =$0F61
   dispSTAT=$0F68      INDshow =$0FCC      FSU     =$0FEB      dispreg =$0FF5
   FMU     =$1000      dispdig =$1033      db      =$1064      bfstart =$1064
   bfptr   =$1066      bfend   =$1068      bfsiz   =$106A      bffn    =$106C
   bfoff   =$106D      bflane  =$1070      bfdirty =$1071      clearAR =$1090
   FDV     =$109B      classdbx=$10B8      fnxdbx  =$10BE      fnxfn   =$10C6
   fnames  =$1100      IFL     =$111E   V  ]fetch4 =$1161      DFL     =$1164
V  ]errpt  =$1171      DLB     =$1174   V? ]dfl    =$1189   V? ]clc    =$11BD
   iosel   =$11C8   V? ]adc    =$11CC   V? ]cmp    =$11CE      setptr  =$11DB
   iodsel  =$11EC   V? ]nop    =$11F6      getwrd  =$11F9   V? ]sub    =$11F9
V  ]incptr6=$120D   V? ]Ov     =$1215      RTF     =$1220   V  ]IOerr1 =$122F
   putwrd  =$1232      readbuf =$1273   V? ]rts    =$127D      nxtblk  =$127E
   IBB     =$1286      DBB     =$1299      incblk  =$12AA      BOF     =$12AC
   BRP     =$12B9      ckpref  =$12BC      BSA     =$12BF      BCH     =$12C9
   prvblk  =$12D2      BCE     =$12DD      BUN     =$12EF      decblk  =$12F6
V? ]fetch3 =$12FF      BFR     =$1302      BFA     =$1306      backoff =$1308
V? ]bfr    =$1308      advoff  =$1328   V  ]resptr =$1340      setlan  =$134B
   BCS     =$1355      SOR     =$1362      resetdbs=$1375      STA     =$1376
   resetdb =$1384      emptydb =$1392   V  ]fetch2 =$13C0      flushall=$13D0
   LDR     =$13DD      flushbuf=$13E3      LDB     =$13E9      dowrite =$13F4
   LSA     =$140F      STP     =$1418   V  ]fetch1 =$142A      CLA     =$142D
V  ]IOerr2 =$142F      doread  =$1432      CLL     =$144E      SRA     =$1459
   PDfae   =$1472      SLA     =$148E      PDebx   =$14AA      bload   =$14D8
   bsave   =$14DF      Aparm   =$14E6      Eparm   =$14EA      Bparm   =$14EE
   putwdhx =$14F2      putbyte =$14F6      srAS    =$1500      srA     =$1502
   srAM    =$1504      srT     =$150D      putpdcmd=$150E      srAMR   =$150F
?  srR     =$1512      srT2    =$151D   ?  pdoscmd =$1520      pdosxeq =$1525
   slT     =$1531      slA     =$153B      AUXcode =$153D      exchAR  =$1546
   splitsL =$1554      clear   =$1568      midNN   =$1575      bcd2bin =$158C
   b220asc =$1626      MTS     =$16B0   V  ]IOerr3 =$172D      MTC     =$1730
   mtbfsz  =$17AC      MRR     =$17AC      MRD     =$17AD      strDinc =$1889
   MIR     =$189C      MIW     =$189D      MOR     =$191A      MOW     =$191B
   MPF     =$1976      MIB     =$19B3      BCDLadrl=$19C7      BCDLadrh=$1A61
   BCDHadrl=$1AFB      BCDHadrh=$1B45      simend  =$1B8E      AUXend  =$27F5
   ptrdr0bf=$3B4C      ptrdr1bf=$3DA6      MEM     =$4AD0      ptpch0bf=$6000
   ptpch1bf=$625A      mt0bf   =$64B4      memb    =$7530      mt1bf   =$7C62
   noAD    =$8000      operr   =$89C3   MD align   =$8000   MD resi    =$8000
MD seti    =$8000   MD mainjsr =$8000   MD mainjmp =$8000   MD auxjsr  =$8000
MD auxjmp  =$8000      DOSCMD  =$BE03   ?  PRINTERR=$BE0C      BSSTATE =$BE42
   KBD     =$C000      READMAIN=$C002      READAUX =$C003      WRITMAIN=$C004
   WRITAUX =$C005      ALTCHAR =$C00F      KBSTROBE=$C010      SPKR    =$C030
   PRBL2   =$F94A   ?  TABV    =$FB5B      BASCALC =$FBC1      BEEP    =$FBDD
   HOME    =$FC58      CROUT   =$FD8E      COUT    =$FDED
```