Tom's
*.asm
old - not
yet

```
                        .INSERT A:S.ASM
                       @.REMARK /
                       @                      *
                       @                    * * *
                       @                     ***
                       @                    *******
                       @                       *
                       @                      *
                       @                     *
                       @                    *
                       @                   *
                       @                  *******
                       @
                       @      "WHEN YOU CARE ENOUGH TO PROGRAM
                       @                 THE VERY BEST"
                       @
                       @              ZGRASS V2.00000000
                       @BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
                       @                   (C) 1978
                       @
                       @/
                        .INSERT A:MAC.ASM
                       @
                        .INSERT A:ZRAM.ASM
                                .LINK
                                .IDENT  ALLOC
                                .PREL
                                .EXTERN ERRPGM
                                .INTERN ALLOC
                                .INTERN ALLOCD
                                .INTERN CORCHK
                                .INTERN CORINI
                                .INTERN DIV16
                                .INTERN FREE
                                .INTERN FREEALL
0003                    N.LNK   = $NLINK
                        ;++++
                        ;
                        ;
                        ; DIV16
                        ; DIVIDES HL BY 16
                        ; CLOBBERS THE ACCUMULATOR!
                        ;
                        ;----
0000'                   DIV16:
0000'  7D                       MOV     A,L             ;A=LOW ORDER
0001'  CB3C                     SRLR    H               ;H=HI ORDER
0003'  1F                       RAR
0004'  CB3C                     SRLR    H
0006'  1F                       RAR
0007'  CB3C                     SRLR    H
0009'  1F                       RAR
000A'  CB3C                     SRLR    H
000C'  1F                       RAR
000D'  6F                       MOV     L,A
```

```
000E'  C9                      RET
                        ;
                        ; CHAIN
                        ; CHAINS THE GIVEN BLOCK AFTER ANOTHER BLOCK IN A
                        ; CIRCULAR LINKED LIST
                        ;
                        ; NEEDS:
                        ;   HL -> BLOCK AFTER WHICH NEW BLOCK GOES
                        ;   DE -> NEW BLOCK TO BE CHAINED IN
                        ;   THE ABOVE REGISTERS POINT TO THE FORWARD POINTER
                        ;   OF THEIR RESPECTIVE BLOCKS, THEY ARE ASSUMED
                        ;   TO BE 2-BYTE ADDRESSES
                        ;
                        ; RETURNS:
                        ;   HL -> NEW BLOCK (JUST CHAINED IN)
                        ;   DE IS DESTROYED, DOES NOT CHANGE A OR Z&C BITS
                        ;
                        ; LOGIC:
                        ;   FWD(DE) = FWD(HL)
                        ;   FWD(HL) = DE
                        ;
                        ; ON ENTRY:
                        ;  HL->******-->******-->?  DE-->******
                        ;       * A *    * B *            * C *
                        ;       ******   ******           ******
                        ;
                        ; ON EXIT:
                        ;       ******-->******-->******-->?
                        ;       * A *    * C *    * B *
                        ;       ******   ******   ******
                        ;
                        ;----
000F'                   CHAIN:
000F'  C5                       PUSH    B               ;LDI USES IT
0010'  D5                       PUSH    D
0011'  E5                       PUSH    H               ;SAVE A PTR
0012'  EDA0                     LDI                     ;(DE) = (HL) FOR 2 BYTES

0014'  EDA0                     LDI                     ;FWD(C) = FWD(A) [B]
0016'  E1                       POP     H               ;DE->C
0017'  D1                       POP     D               ;HL->A
0018'  73                       MOV     M,E             ;(HL) = DE
0019'  23                       INX     H
001A'  72                       MOV     M,D             ;FWD(A) = C
001B'  EB                       XCHG                    ;HL->NEW BLOCK C
001C'  C1                       POP     B
001D'  C9                       RET
                        ;++++
                        ;
                        ; DECHAIN
                        ; CHAINS THE SPECIFIED BLOCK OUT OF A CIRCULAR
                        ; LINKED LIST, DOES NOT CHECK IF IT IS CHAINING
                        ; OUT THE HEADER
                        ;
                        ; NEEDS:
```

```
                        ;    HL -> BLOCK BEFORE BLOCK TO BE CHAINED OUT
                        ;         (PREDECESSOR)
                        ;
                        ; RETURNS:
                        ;    HL -> BLOCK CHAINED OUT
                        ;    DOES NOT CHANGE A OR Z&C BITS
                        ;
                        ; IT IS ASSUMED THAT THE FORWARD POINTER TO THE
                        ; GIVEN BLOCK POINTS TO THE FORWARD POINTER OF THE
                        ; BLOCK TO BE CHAINED OUT
                        ; ON ENTRY:
                        ;  HL-->******-->******-->******
                        ;       * A *    * B *    * C *
                        ;       ******   ******   ******
                        ;
                        ; ON EXIT:
                        ;       ******-->******       HL-->******
                        ;       * A *    * B *             * C *
                        ;       ******   ******            ******
                        ;
                        ;----
001E'                   DECHAIN:
001E'  C5                       PUSH    B               ;LDI USES IT
001F'  D5                       PUSH    D
0020'  5E                       MOV     E,M             ;DE = FWD(HL)
0021'  23                       INX     H
0022'  56                       MOV     D,M
0023'  2B                       DCX     H               ;HL->A AGAIN
0024'  EB                       XCHG                    ;HL->B, DE->A
0025'  E5                       PUSH    H
0026'  EDA0                     LDI                     ;(DE) = (HL) FOR 2 BYTES

0028'  EDA0                     LDI                     ;FWD(A) = FWD(B)
002A'  E1                       POP     H               ;HL->B (LINKED OUT)
002B'  D1                       POP     D
002C'  C1                       POP     B
002D'  C9                       RET
                        ;++++
                        ;
                        ; ALLOC
                        ; ALLOCATES A BLOCK (LARGEST OR GIVEN SIZE) OF
                        ; MEMORY FROM THE FREE LIST
                        ;
                        ; NEEDS:
                        ;    A = NUMBER OF BLOCKS TO ALLOCATE (0-254 -> 1-255)
                        ;         IF -1, ALLOCATE THE LARGEST BLOCK
                        ;
                        ; RETURNS:
                        ;    HL->TOP OF NEW BLOCK
                        ;    BC = SIZE (IN BYTES) OF NEW BLOCK (0-BASED)
                        ;    HL->*********
                        ;         *   0   * TYPE
                        ;         * 1-255 * SIZE
                        ;         *       * REST OF BLOCK
                        ;         *********
```

```
                        ; ERRORS:
                        ;   ER.COR  - OUT OF MEMORY
                        ;
                        ; LIST LOOKS LIKE:
                        ;       FREELST
                        ;       *********       *********
                        ;       * 0     *       * 0     *
                        ;       * 0     *       * SIZE  *
                        ;       *PTR TO *------>*PTR TO *--->?
                        ;       *LARGEST*       *NEXT   *
                        ;       *********       *********
                        ;
                        ; THE FREE LIST IS MAINTAINED AS A CIRCULAR
                        ; LINKED LIST OF FREE CHUNKS SORTED BY SIZE
                        ; (LARGEST FIRST). WHEN IT IS EMPTY, THE LISTHEAD
                        ; POINTS TO IT'S OWN FORWARD POINTER
                        ;
                        ;----
002E'                   ALLOC:
002E'   D5                      PUSH    D
                        ;+
                        ; WE POINT HL AT THE LISTHEAD FORWARD PTR. THIS IS
                        ; THE PREDECESSOR TO THE BLOCK TO ALLOCATE FOR UNKNOWN
                        ; SIZE ALLOCATION AND THE STARTING PREDECESSOR FOR
                        ; THE LIST SCAN FOR UNKNOWN SIZE ALLOCATION. IF A
                        ; IS -1, WE ALLOCATE THE LARGEST (FIRST IN THE LIST)
                        ;-
002F'   21 65E5                 LXI     H,FREELST       ;HL->FREE LISTHEAD
0032'   23                      INX     H               ;HL->TYPE FIELD
0033'   23                      INX     H               ;HL->FORWARD PTR
0034'   E5                      PUSH    H               ;SP->PRECECESSOR PTR
0035'   5E                      MOV     E,M             ;DE -> FIRST BLOCK
0036'   23                      INX     H               ;IN LIST, IN CASE
0037'   56                      MOV     D,M             ;OF UNKNOWN SIZE
0038'   2B                      DCX     H               ;ALLOCATION
0039'   3C                      INR     A               ;KNOWN SIZE?
003A'   47                      MOV     B,A             ;B=SIZE WE WANT
003B'   200B                    JRNZ    ..A2            ;YES? SEARCH LIST
003D'   EB                      XCHG                    ;HL->FWD PTR
003E'   2B                      DCX     H               ;HL->SIZE FIELD
003F'   46                      MOV     B,M             ;B=SIZE OF BLOCK
0040'   1810                    JMPR    ..A4
                        ;+
                        ; SP -> PTR TO THE PREDECESSOR OF CURRENT BLOCK
                        ; HL -> CURRENT BLOCK (PREVIOUS ONE LARGE ENOUGH)
                        ; DE -> FOLLOWER (ONE JUST CHECKED FITS)
                        ; B = SIZE (IN BLOCKS) OF REQUEST
                        ; WE MUST KEEP TRACK OF THE PREDECESSOR TO THE
                        ; CURRENT BLOCK FOR THE DECHAIN OPERATION (THIS IS
                        ; A SINGLY LINKED LIST). HL POINTS TO THE BLOCK
                        ; WHICH IS A CANDIDATE FOR ALLOCATION. WHEN IT'S
                        ; FOLLOWER HAS A SIZE TOO SMALL THEN THIS BLOCK IS
                        ; THE BEST FIT BLOCK. THE LIST IS SCANNED IN ORDER
                        ; UNTIL A BEST FIT IS FOUND, THE ORDER OF THE BLOCKS
```

```
                        ; ARE LARGEST FIRST
                        ;-
0042'  E3               ..A1:   XTHL                    ;PREDECESSOR = CURRENT
0043'  EB                       XCHG                    ;CURRENT = FOLLOWER
0044'  5E                       MOV     E,M             ;DE -> NEW FOLLOWER
0045'  23                       INX     H
0046'  56                       MOV     D,M
0047'  2B                       DCX     H               ;HL -> CURRENT
0048'  1B               ..A2:   DCX     D               ;DE -> SIZE OF FOLLOWER
0049'  1A                       LDAX    D               ;A=SIZE OF FOLLOWER
                                TST     A               ;END OF LIST?[
004A'  B7               +       ORA     A]
004B'  2804                     JRZ     ..A3            ;YES? USE CURRENT ONE
004D'  B8                       CMP     B               ;CURRENT TOO BIG?
004E'  13                       INX     D               ;(DE), WE LOOP AGAIN
004F'  30F1                     JRNC    ..A1            ;(DE) >= B
                        ;+
                        ; HL -> FORWARD POINTER OF BLOCK TO ALLOCATE
                        ; SP -> POINTER TO PREDECESSOR OF CURRENT BLOCK
                        ; A = SIZE (IN BLOCKS) TO ALLOCATE
                        ; IT IS POSSIBLE THAT THE BLOCK WE FOUND IS ACTUALLY
                        ; THE LISTHEAD, THAT IS THERE IS NO BLOCK WHICH
                        ; IS LARGE ENOUGH FOR USE (OR NO BLOCKS AT ALL). IF
                        ; THE SIZE OF THE BLOCK IS 0, ITS THE LISTHEAD, OUT
                        ; OF MEMORY
                        ;-
0051'  2B               ..A3:   DCX     H               ;HL->SIZE OF NEW BLOCK
0052'  5E               ..A4:   MOV     E,M             ;E = SIZE OF BLOCK
                                TST     E               ;SIZE = 0?[
0053'  7B               +.IFN E         -A, [   MOV     A,E             ]
0054'  B7               +       ORA     A]
                                ZERROR  ER.COR          ;YES? OUT OF CORE![
0055'  2004             +       JRNZ ..0001
0057'  CD 0000:04       +       CALL ERRPGM
005A'  1B               +       .BYTE ER.COR
005B'                   +..0001:]
005B'  78                       MOV     A,B             ;A = SIZE WE REQUESTED
                        ;+
                        ; HL -> SIZE FIELD OF BLOCK TO ALLOCATE
                        ; A = SIZE (BLOCKS) WE REQUESTED
                        ; E = SIZE (BLOCKS) WE ALLOCATED
                        ; NOW WE COMPUTE THE NUMBER OF BLOCKS LEFT OVER
                        ; IN THE BLOCK (AFTER WE ALLOCATE THE NUMBER REQUESTED)
                        ; IF THERE ARE ANY, THIS FRAGMENT MUST BE RETURNED
                        ; TO THE FREE LIST
                        ;-
005C'  77                       MOV     M,A             ;SIZE = AMOUNT WANTED
                                COM                     ;NEGATE # OF BLOCKS[
005D'  2F               +       CMA
005E'  3C               +       INR     A]
005F'  83                       ADD     E               ;A = # BLOCKS OVER
0060'  5E                       MOV     E,M             ;E = BLOCKS ALLOCATED
                        ;+
                        ; A = NUMBER OF BLOCKS LEFT OVER
                        ; DE = NUMBER OF BLOCKS ACTUALLY ALLOCATED
```

```
                        ; SP -> POINTER TO PREDECESSOR OF ALLOCATED BLOCK
                        ; WE MUST CONVERT THE BLOCK SIZE ALLOCATED TO A
                        ; BYTE SIZE AND SAVE IN BC FOR THE USER. THEN THE
                        ; BLOCK IS CHAINED OUT OF THE FREE LIST
                        ;-
0061'   EB                      XCHG                    ;HL = BLOCK SIZE
                                CLR     H[
0062'   2600           +.IFN H-A, [    MVI     H,0][   XRA     A]]
0064'   29                      DAD     H               ;HL = BYTE SIZE
0065'   29                      DAD     H               ;H =* 16
0066'   29                      DAD     H
0067'   29                      DAD     H
                                MVD     B,H             ;BC = BYTE SIZE[
0068'   44              +       MOV     B,H
0069'   4D              +       MOV     B+1,H           +1]
006A'   E1                      POP     H               ;HL -> PREDECESSOR
006B'   CD 001E'                CALL    DECHAIN         ;OUT OF FREE LIST
                        ;+
                        ; A = NUMBER LEFT OVER (CC'S STILL SET AT THIS POINT)
                        ; HL -> TOP OF CURRENT BLOCK ALLOCATED
                        ; WE CHAINED THE CURRENT BLOCK OUT OF THE LIST,
                        ; NOW WE GARBAGE COLLECT THE FRAGMENT (IF ANY)
                        ; THE BYTE SIZE IS ADDED TO THE POINTER TO GET THE
                        ; ADDRESS OF THE SIZE FIELD OF THE FRAGMENT
                        ;-
006E'   2809                    JRZ     ..EX            ;NO FRAGMENT? RETURN
0070'   E5                      PUSH    H               ;SAVE OUR ADDR
0071'   09                      DAD     B               ;HL -> FRAGMENT SIZE
0072'   2B                      DCX     H               ;HL -> SIZE FIELD
0073'   77                      MOV     M,A             ;SIZE = # BLOCKS OVER
0074'   2B                      DCX     H               ;HL -> TOP OF BLOCK
0075'   CD 00AD'                CALL    FREE            ;FREE FRAGMENT
0078'   E1                      POP     H               ;HL->BLOCK ALLOCD
0079'   2B              ..EX:   DCX     H               ;HL->SIZE
007A'   2B                      DCX     H               ;HL->TOP
007B'   D1                      POP     D               ;RETURN
007C'   C9                      RET
                        ;++++
                        ;
                        ; CHKREC
                        ; CHECKS IF THE GIVEN BLOCKS CAN BE RECOMBINED
                        ; AND RETURNS NEW SIZE IF SO
                        ;
                        ; NEEDS:
                        ;   HL -> SIZE FIELD OF BLOCK TO BE ADDED TO
                        ;   DE -> SIZE FIELD OF BLOCK TO BE RECOMBINED WITH
                        ;
                        ; RETURNS:
                        ;   IF RECOMBINATION IS POSSIBLE:
                        ;       A = COMBINED SIZE OF NEW BLOCK (1-255)
                        ;       CC'S CLEAR
                        ;       ELSE C BIT SET, A DESTROYED
                        ;
                        ; ON ENTRY:
                        ;       **********      **********
```

```
                        ;        *  ...   *      *  ...   *
                        ;  HL-->* SIZE A *      * SIZE B *
                        ;        *  ...   *      *  ...   *
                        ;        **********      **********
                        ;
                        ; CHKREC DOES NOT MODIFY ANY FIELDS, CHECKS TO SEE
                        ; IF THE BLOCK DE POINTS AT DIRECTLY FOLLOWS THE
                        ; BLOCK HL POINTS AT IN MEMORY (THAT IS, IT CHECKS
                        ; IF HL + SIZE A = DE (DOES NOT CHECK IF DE + SIZE B
                        ; = HL)
                        ;
                        ;-----
007D'                   CHKREC:
007D'  E5                       PUSH    H               ;SAVE OUR REGS
007E'  D5                       PUSH    D
                        ;+
                        ; IF HL + (DE) = DE THEN THE BLOCK DE POINTS AT
                        ; DIRECTLY FOLLOWS THE BLOCK HL POINTS AT AND THEY
                        ; CAN BE PHYSICALLY RECOMBINED
                        ;-
007F'  5E                       MOV     E,M             ;DE=SIZE OF BLOCK A
                                CLR     D[
0080'  1600             +.IFN D-A, [    MVI     D,0][   XRA     A]]
0082'  EB                       XCHG
0083'  29                       DAD     H               ;CONVERT FROM BLOCK
0084'  29                       DAD     H               ;SIZE TO BYTE SIZE
0085'  29                       DAD     H
0086'  29                       DAD     H
0087'  EB                       XCHG
0088'  19                       DAD     D               ;HL -> END OF BLOCK A
0089'  D1                       POP     D               ;DE -> SIZE OF BLOCK B
008A'  7A                       MOV     A,D             ;IF HL = DE THEN BLOCK B

008B'  BC                       CMP     H               ;FOLLOWS BLOCK A AND
008C'  C2 00AA'                 JNZ     ..ERR           ;THEY CAN BE RECOMBINED
008F'  7B                       MOV     A,E
0090'  BD                       CMP     L
0091'  2017                     JRNZ    ..ERR
                        ;+
                        ; WE KNOW THEY CAN BE RECOMBINED PHYSICALLY BUT SINCE
                        ; THE SIZE IS ONLY ONE BYTE IT IS POSSIBLE THE SUM OF
                        ; THEIR SIZES WILL BE TOO BIG TO FIT IN 1 BYTE AND
                        ; THEN WE CAN'T RECOMBINE THEM. ALSO SIZES OVER
                        ; THE LIMIT OF MAXFRG ARE ILLEGAL
                        ;-
0093'  E1                       POP     H               ;HL -> SIZE OF BLOCK A
0094'  E5                       PUSH    H               ;SAVE IT
0095'  D5                       PUSH    D               ;AND SAVE SIZE OF B
0096'  1A                       LDAX    D               ;SAVE BLOCK B SIZE
                                CLR     D               ;DE = SIZE OF BLOCK A[
0097'  1600             +.IFN D         -A, [   MVI     D                ,0][
                        XRA     A]]
0099'  5E                       MOV     E,M
009A'  62                       MOV     H,D             ;HL = SIZE OF BLOCK B
009B'  6F                       MOV     L,A
```

```
009C'  19                      DAD     D               ;HL = SIZE A + SIZE B
009D'  D1                      POP     D               ;D OFF STACK
                               TST     H               ;IF H != 0,SIZE[
009E'  7C              +.IFN H         -A, [   MOV     A,H                 ]
009F'  B7              +       ORA     A]
00A0'  2008                    JRNZ    ..ERR           ;IS TOO BIG
00A2'  7D                      MOV     A,L             ;A = SIZE A + SIZE B
00A3'  FE41                    CPI     MAXFRG+1        ;SIZE > MAXFRG?
00A5'  3003                    JRNC    ..ERR           ;FAIL THEN
                               CLC                     ;SUCCEED[
00A7'  B7              +       ORA     A]
00A8'  E1                      POP     H
00A9'  C9                      RET
00AA'  E1               ..ERR: POP     H
00AB'  37                      STC
00AC'  C9                      RET
                       ;++++
                       ;
                       ; FREE
                       ; RETURNS A BLOCK OF MEMORY TO THE FREE-CORE LIST,
                       ; CAN BE USED AGAIN IF A CALL TO ALLOC IS DONE
                       ;
                       ; NEEDS:
                       ;   HL -> TOP OF BLOCK TO BE RETURNED
                       ;       **********
                       ;       * TYPE   * 0 FOR FREE BLOCKS
                       ;       * SIZE   * 1-255 (# OF 16 BYTE BLOCKS)
                       ;       *        *
                       ;       **********
                       ;
                       ; RETURNS:
                       ;   SUCCEED:
                       ;
                       ;       SUCCEEDS IF BLOCK COULD BE RETURNED TO FREE LIST

                       ;   FAIL:
                       ;       CARRY BIT SET, FAILS IF THE PTR TO BLOCK
                       ;       TO FREE WAS 0 UPON ENTRY (CANNOT FREE 0)
                       ;
                       ; CALLS:
                       ;   CHKREC - CHECKS FOR RECOMBINATION
                       ;   CHAIN, DECHAIN - LIST MANAGEMENT
                       ;
                       ; THE FREE-CORE LIST IS A CIRCULAR LINKED LIST SORTED
                       ; BY SIZE (LARGEST FIRST) WHEN A BLOCK IS FREED,
                       ; IT MAY BE RECOMBINED WITH THE BLOCK BEFORE OR
                       ; AFTER IT IN PHYSICAL MEMORY IF THE SUM OF THEIR
                       ; SIZES IS NOT GREATER THAN 254 (SEE CHKREC,ALLOC)
                       ;
                       ;-----
00AD'                  FREE:
00AD'  7C                      MOV     A,H             ;IF HL = 0, FAIL
00AE'  B5                      ORA     L
00AF'  2852                    JRZ     ..FAIL
00B1'  E5                      PUSH    H               ;SAVE ALL REGS
```

```
00B2'   D5                      PUSH    D
00B3'   C5                      PUSH    B
                                CLR     M                       ;CLEAR TYPE[
00B4'   3600            +.IFN M         -A, [   MVI     M               ,0][
                         XRA    A]]
00B6'   23                      INX     H                       ;HL -> SIZE OF BLOCK
00B7'   EB                      XCHG                            ;DE -> TOP OF BLOCK
00B8'   21 65E5          ..F0:  LXI     H,FREELST               ;HL -> LISTHEAD
00BB'   23                      INX     H                       ;HL -> SIZE OF LISTHEAD
                        ;+
                        ; BC -> FWD PTR OF PREDECESSOR (BLOCK BEFORE
                        ; CURRENT BLOCK IN LIST)
                        ; HL -> SIZE OF CURRENT BLOCK JUST CHECKED
                        ; DE -> SIZE OF BLOCK TO FREE
                        ; WE NOW GET THE NEXT BLOCK IN THE LIST TO CHECK,
                        ; WE MUST SAVE THE ADDRESS OF THE OLD BLOCK
                        ; (PREDECESSOR). WE CHECK RECOMBINATION IN THIS
                        ; LOOP (BOTH KINDS) IF A BLOCK MUST BE RECOMBINED
                        ; IT IS PULLED OUT OF THE LIST AND FREE IS CALLED
                        ; AGAIN TO PLACE IT PROPERLY
                        ;-
00BC'   23               ..F1:  INX     H                       ;HL -> FWD PTR
                                MVD     B,H                     ;BC = OLD BLOCK'S ADDR[
00BD'   44              +       MOV     B,H
00BE'   4D              +       MOV     B+1,H           +1]
00BF'   D5                      PUSH    D                       ;SAVE NEW BLOCK PTR
00C0'   5E                      MOV     E,M                     ;DE -> NEXT BLOCK
00C1'   23                      INX     H
00C2'   56                      MOV     D,M
00C3'   EB                      XCHG                            ;HL -> FP OF NEXT BLK
00C4'   D1                      POP     D                       ;DE -> NEW BLOCK
00C5'   2B                      DCX     H                       ;HL -> SIZE OF FOLLOWER
                                TST     M                       ;A=SIZE, 0?[
00C6'   7E              +.IFN M         -A, [   MOV     A,M             ]
00C7'   B7              +       ORA     A]
00C8'   2818                    JRZ     ..F6                    ;NOW TRY TO INSERT IT
                        ;+
                        ; BC -> FWD PTR OF PREDECESSOR BLOCK
                        ; HL -> SIZE OF CURRENT BLOCK
                        ; IF THE 2 BLOCKS CAN BE RECOMBINED (CHKREC TELLS
                        ; US THIS) THEN WE TAKE THE CURRENT BLOCK OUT OF
                        ; THE LIST AND PUT THE NEW COMBINED BLOCK BACK
                        ; IN ITS PROPER PLACE (THE LIST IS SORTED BY SIZE
                        ; SO IT IS VERY LIKELY THAT IT WILL GO FURTHER BACK)
                        ;-
00CA'   CD 007D'                CALL    CHKREC                  ;CAN WE RECOMBINE?
00CD'   380A                    JRC     ..F2                    ;NO? CHECK OTHER WAY
00CF'   77               ..F4:  MOV     M,A                     ;SAVE NEW SIZE
00D0'   E5                      PUSH    H                       ;SAVE CURRENT PTR
                                MVD     H,B                     ;HL -> FP OF PRED.[
00D1'   60              +       MOV     H,B
00D2'   69              +       MOV     H+1,B           +1]
00D3'   CD 001E'                CALL    DECHAIN                 ;PUT IT BACK
00D6'   D1                      POP     D                       ;DE->BLOCK TO FREE
00D7'   18DF                    JMPR    ..F0                    ;WHERE IT BELONGS
```

```
                        ;+
                        ; HL -> SIZE OF CURRENT BLOCK
                        ; DE -> SIZE OF BLOCK TO FREE
                        ; NOW WE CHECK RECOMBINATION IN THE OTHER DIRECTION,
                        ; WE SEE IF THE BLOCK DE POINTS AT COMES PHYSICALLY
                        ; IMMEDIATELY BEFORE THE BLOCK HL POINTS AT.
                        ; IF SO, WE CHAIN OUT THE CURRENT BLOCK AND FREE
                        ; THE RECOMBINED BLOCK
                        ;-
00D9'   EB              ..F2:   XCHG                    ;HL -> BLOCK TO FREE
                                                        ;DE -> CURRENT BLOCK
00DA'   CD 007D'                CALL    CHKREC          ;CAN RECOMBINE?
00DD'   30F0                    JRNC    ..F4            ;YES? DECHAIN & FREE
00DF'   EB                      XCHG                    ;RESTORE PTRS
00E0'   18DA                    JMPR    ..F1            ;AND CHECK NEXT ONE
                        ;+
                        ; NO RECOMBINATION IS POSSIBLE, NOW WE MUST PUT
                        ; THE CURRENT BLOCK SOMEWHERE IN THE LIST (WHICH
                        ; IS SORTED BY SIZE) IT WILL GO BEFORE THE FIRST
                        ; BLOCK FOUND WHOSE SIZE IS SMALLER THAN IT'S SIZE
                        ;-
00E2'   21 65E5         ..F6:   LXI     H,FREELST       ;HL -> LISTHEAD
00E5'   23                      INX     H               ;HL -> SIZE
00E6'   23              ..F7:   INX     H               ;HL -> FWD PTR
                                MVD     B,H             ;BC = OLD HL[
00E7'   44              +       MOV     B,H
00E8'   4D              +       MOV     B+1,H           +1]
00E9'   D5                      PUSH    D               ;SAVE NEW BLOCK PTR
00EA'   5E                      MOV     E,M             ;DE -> FP OF NEXT ITEM
00EB'   23                      INX     H
00EC'   56                      MOV     D,M
00ED'   1B                      DCX     D               ;DE -> SIZE FIELD
00EE'   EB                      XCHG                    ;HL -> CURRENT LIST ITEM

00EF'   D1                      POP     D               ;DE -> SIZE OF BLOCK
                        ;+
                        ; IF THE SIZE OF THE BLOCK TO FREE IS > THE SIZE
                        ; OF THE CURRENT BLOCK THEN IT IS TO BE INSERTED
                        ; IN THE LIST BEFORE THE CURRENT BLOCK (NOTE THAT
                        ; IF WE DIDN'T HAVE THE FORESIGHT TO SAVE THE PTR
                        ; TO THE PREDECESSOR OF THE CURRENT BLOCK WE WOULD
                        ; BE IN TROUBLE HERE)
                        ;-
                                TST     M               ;A=SIZE, IF 0 END[
00F0'   7E              +.IFN M         -A, [   MOV     A,M             ]
00F1'   B7              +       ORA     A]
00F2'   2804                    JRZ     ..F5            ;PUT IN LIST THEN
00F4'   1A                      LDAX    D               ;COMPARE THE SIZES
00F5'   BE                      CMP     M               ;A > (HL)? DONT GO HERE
00F6'   38EE                    JRC     ..F7
00F8'                   ..F5:   MVD     H,B             ;HL = BC -> PREDECESSOR[

00F8'   60              +       MOV     H,B
00F9'   69              +       MOV     H+1,B           +1]
00FA'   13                      INX     D               ;DE->FWD PTR FIELD
```

```
00FB'  CD 000F'            CALL    CHAIN           ;CHAIN IT IN
                           CLC                     ;CLEAR CARRY, WE FREED[
00FE'  B7          +       ORA     A]
00FF'  C1                  POP     B
0100'  D1                  POP     D
0101'  E1                  POP     H
0102'  C9                  RET
0103'  37           ..FAIL: STC                    ;FAIL
0104'  C9                  RET
                   ;++++
                   ;
                   ; ALLOCD
                   ; RETURNS A FRAGMENT OF A BLOCK TO THE FREE MEMORY LIST,

                   ; UPDATES THE SIZE OF THE ORIGINAL BLOCK
                   ;
                   ; NEEDS:
                   ;   HL -> TOP OF ORIGINAL BLOCK
                   ;   DE -> PAST LAST BYTE USED IN THE BLOCK
                   ;
                   ; RETURNS:
                   ;   REGS SAME AS UPON ENTRY
                   ;   SIZE FIELD OF BLOCK IS THE SIZE OF THE USED PART OF
                   ;   THE BLOCK, FRAGMENT NOW IN FREE LIST
                   ;
                   ; CALLS:
                   ;   FREE - RETURNS FRAGMENT TO FREE LIST
                   ;   DIV16 - DIVIDES HL BY 16
                   ;
                   ;----
0105'              ALLOCD:
0105'  C5                  PUSH    B               ;SAVE REGS
0106'  D5                  PUSH    D
0107'  E5                  PUSH    H
                   ;+
                   ; DE -> LAST BYTE USED IN THE BLOCK
                   ; HL -> TOP OF BLOCK
                   ; WE MUST FIND THE NUMBER OF BYTES WHICH WERE USED AND
                   ; CONVERT THIS TO BLOCKS, DE - HL GIVES THE # OF BYTES
                   ;-
0108'  E5                  PUSH    H               ;SAVE TOP PTR
0109'  D5                  PUSH    D               ;SAVE END PTR
010A'  EB                  XCHG                    ;GET DE - HL
                           CLC                     ;CLEAR THE CARRY[
010B'  B7          +       ORA     A]
010C'  ED52                DSBC    D               ;HL =- DE
010E'  CD 0000'            CALL    DIV16           ;HL =/ 16 (->BLOCKS)
0111'  7D                  MOV     A,L             ;A = # OF BLOCKS USED
0112'  3C                  INR     A               ;MAKE 1 BASED
                   ;+
                   ; A = # OF BLOCKS USED
                   ; DE -> PAST LAST BYTE USED
                   ; WE MUST NOW GET THE ADDRESS OF THE NEXT BLOCK AFTER
                   ; THE LAST ONE USED, THIS WILL BE THE LEFTOVER FRAGMENT
                   ; DE IS ROUNDED UP TO THE NEXT BLOCK
```

```
                        ;-
0113'  D1                       POP     D               ;DE -> END
0114'  F5                       PUSH    PSW             ;SAVE OUR SIZE
0115'  21 0010                  LXI     H,16            ;HL = BLOCK SIZE
0118'  19                       DAD     D               ;HL = DE + BLOCKSIZE
0119'  7D                       MOV     A,L             ;L = LO ORDER BYTE
011A'  E6F0                     ANI     0F0H            ;MASK LO ORDER 4 BITS
011C'  6F                       MOV     L,A             ;HL -> FOLLOWING BLOCK
011D'  EB                       XCHG                    ;DE -> FOLLOWING BLOCK
011E'  F1                       POP     PSW             ;RESTORE BLOCKS USED
011F'  E1                       POP     H               ;HL -> TOP OF BLOCK
0120'  23                       INX     H               ;HL -> SIZE OF BLOCK
0121'  4E                       MOV     C,M             ;HL -> SIZE OF ORIGINAL
0122'  77                       MOV     M,A             ;SAVE SIZE OF USED BLOCK

                                COM                     ;A = -USED[
0123'  2F               +       CMA
0124'  3C               +       INR     A]
0125'  81                       ADD     C               ;A = ORIGINAL-USED
0126'  2807                     JRZ     ..EX            ;0? NO FRAGMENT, EXIT
0128'  13                       INX     D               ;DE -> SIZE OF FRAGMENT
0129'  12                       STAX    D               ;SAVE FRAGMENT SIZE
012A'  1B                       DCX     D               ;DE -> TOP AGAIN
012B'  EB                       XCHG                    ;HL -> TOP FOR FREE
012C'  CD 00AD'                 CALL    FREE            ;FREE THE LEFTOVER PART
012F'  E1               ..EX:   POP     H               ;RESTORE REGS
0130'  D1                       POP     D
0131'  C1                       POP     B
0132'  C9                       RET
                        ;++++
                        ;
                        ; FREEALL
                        ; FREES THE ELEMENTS OF A LINKED LIST
                        ;
                        ; NEEDS:
                        ;   HL -> TOP OF FIRST BLOCK IN LIST TO BE FREED
                        ;         N.LNK(HL) -> TOP OF FOLLOWING BLOCK TO BE FREED
                        ;         IF 0, END OF LIST
                        ;
                        ; HL -> **********   --->**********
                        ;       *  ...  *  /    *  ...  *
                        ; N.LNK *PTR TO *--     *PTR TO *--->REST OF LIST
                        ;       * NEXT  *       * NEXT  *
                        ;       **********      **********
                        ;
                        ; RETURNS:
                        ;   REGS ARE SAME AS UPON ENTRY
                        ;   ALL MEMBERS OF LINKED LIST FREED
                        ;   NOTE - DO NOT TRY TO USE ANY OF THE FIELDS IN
                        ;   THE ELEMENTS AFTER THEY HAVE BEEN FREED! FREE
                        ;   MAY CHANGE THE DATA
                        ;
                        ; CALLS:
                        ;   FREE - FREES A SINGLE ENTRY
                        ;
```

```
                        ;----
0133'                   FREEALL:
0133'   DDE5                   PUSH    X               ;SAVE A REG
0135'   E5                     PUSH    H
0136'   D5                     PUSH    D
0137'   E5              ..F1:   PUSH    H               ;IX = HL
0138'   DDE1                   POP     X
013A'   DD5E03                 MOV     E,N.LNK(X)      ;DE -> NEXT ONE IN LIST
013D'   DD5604                 MOV     D,N.LNK+1(X)    ;HL -> CURRENT ONE
0140'   CD 00AD'               CALL    FREE            ;FREES THE CURRENT ONE
0143'   EB                     XCHG                    ;HL -> NEXT ONE TO FREE
0144'   30F1                   JRNC    ..F1            ;OK TO FREE IT?
0146'   D1                     POP     D
0147'   E1                     POP     H
0148'   DDE1                   POP     X
014A'   C9                     RET
                        ;++++
                        ;
                        ; CORINI
                        ; SETS UP INITIAL MEMORY IN THE FREE LIST
                        ;
                        ; THE CHUNKS ARE 'FRAGSIZ' BLOCKS EACH EXCEPT
                        ; FOR THE LAST ONE WHICH MAY OR MAY NOT BE
                        ; SMALLER. MEMORY IS ASSUMED TO RANGE FROM
                        ; 'RAMSTRT' TO 'RAMEND' AND THE CHUNKS ARE
                        ; PHYSICALLY CONSECUTIVE
                        ;
                        ;----
014B'                   CORINI:
014B'   E5                     PUSH    H
014C'   C5                     PUSH    B
014D'   D5                     PUSH    D
014E'   21 65E7                LXI     H,FWDPTR
0151'   22 65E7                SHLD    FWDPTR          ; INITIALIZE POINTER
0154'   01 016F                LXI     B,(RAMEND-RAMSTRT)/16   ;SIZE
0157'   11 6900                LXI     D,RAMSTRT       ;STARTING ADDR
015A'   21 65E7                LXI     H,FREELST+2     ;HL->FWD PTR OF
015D'   E5                     PUSH    H               ;LISTHEAD, SAVE IT
                        ;+
                        ; DE -> TOP OF CURRENT BLOCK
                        ; SP -> FWD PTR FIELD OF PREDECESSOR
                        ; WE MUST CLEAR THE TYPE OF THE CURRENT BLOCK
                        ; AND STORE IT'S ADDRESS IN THE FWD PTR FIELD
                        ; OF ITS PREDECESSOR (SAVED ON STACK)
                        ;-
                        ..C1:   CLR     A               ;CLEAR TYPE[
015E'   AF              +.IFN A         -A, [   MVI     A                       ,0][
                        XRA     A]]
015F'   12                     STAX    D               ;IN 1ST WORD
0160'   13                     INX     D               ;DE->SIZE FIELD
0161'   13                     INX     D               ;DE->FWD PTR FIELD
0162'   E1                     POP     H               ;HL->FWD PTR OF
0163'   73                     MOV     M,E             ;PREDECESSOR, WILL
0164'   23                     INX     H               ;BECOME ADDR OF
0165'   72                     MOV     M,D             ;CURRENT FP FIELD
```

```
0166'  D5                      PUSH    D               ;PUSH IT FOR FOLLWR
0167'  1B                      DCX     D               ;DE->SIZE FIELD AGAIN
                       ;+
                       ; DE -> SIZE FIELD OF CURRENT BLOCK
                       ; BC = TOTAL SIZE LEFT IN MEMORY
                       ; IF THE TOTAL SIZE - FRAGMENT SIZE IS NEGATIVE,
                       ; THE LAST BLOCK WILL BE SMALLER (NOT A FULL SIZE
                       ; FRAGMENT). OTHERWISE, THE STANDARD FRAGMENT SIZE
                       ; 'FRAGSIZ' IS STORED IN THE SIZE FIELD AND THE
                       ; ADDRESS OF THE NEXT BLOCK IS COMPUTED BY ADDING
                       ; THE FRAGMENT SIZE (IN BYTES) TO THE CURRENT ADDR
                       ;-
0168'  21 FFC0                 LXI     H,-(FRAGSIZ/16) ;HL=-BLOCKSIZE
016B'  09                      DAD     B               ;HL=BLOCKS LEFT
                               TST     H               ;IF -, FRAG IS LESS[
016C'  7C              +.IFN H         -A, [   MOV     A,H             ]
016D'  B7              +               ORA     A]
016E'  FA 0181'                JM      ..EX            ;WE USE SMALLER SIZE
0171'  B5                      ORA     L               ;IF 0, ALSO EXIT
0172'  280D                    JRZ     ..EX            ;WITH SMALLER
                               MVD     B,H             ;BC = SIZE LEFT[
0174'  44              +       MOV     B,H
0175'  4D              +       MOV     B+1,H           +1]
0176'  3E40                    MVI     A,FRAGSIZ/16    ;A=BLOCK SIZE
0178'  12                      STAX    D               ;OF FRAGMENT
0179'  1B                      DCX     D               ;BACK TO TOP
017A'  21 0400                 LXI     H,FRAGSIZ       ;BYTE SIZE OF FRAG
017D'  19                      DAD     D               ;NEW ADDR IN HL
017E'  EB                      XCHG                    ;HL->NEXT BLOCK
017F'  18DD                    JMPR    ..C1            ;GET ITS SIZE
                       ;+
                       ; DE -> SIZE FIELD OF CURRENT BLOCK
                       ; BC = SIZE OF CURRENT BLOCK - 'FRAGSIZ'
                       ; WE ADD THE STANDARD FRAGMENT SIZE BACK TO
                       ; BC TO GET THE SIZE LEFT AND THIS IS THE SIZE
                       ; OF OUR LAST FRAGMENT. ITS FORWARD POINTER
                       ; WILL BE TO THE FORWARD POINTER FIELD OF THE LISTHEAD
                       ;-
0181'  C1              ..EX:   POP     B
0182'  01 0040                 LXI     B,FRAGSIZ/16    ;ADD TO GET SIZE
0185'  09                      DAD     B               ;OF SMALLER IN HL
0186'  7D                      MOV     A,L             ;A=BLOCK SIZE
0187'  12                      STAX    D               ;SAVE IN LAST ONE
0188'  13                      INX     D               ;DE -> FWD PTR FIELD
0189'  21 65E7                 LXI     H,FREELST+2     ;HL->FWD PTR OF FREELST
018C'  EB                      XCHG                    ;HL->FIELD,DE=VALUE
018D'  73                      MOV     M,E             ;SAVE THE FWD PTR
018E'  23                      INX     H               ;IN THE FIELD OF
018F'  72                      MOV     M,D             ;LAST BLOCK, CIRCULAR
0190'  D1                      POP     D               ;RESTORE REGS
0191'  C1                      POP     B
0192'  E1                      POP
0193'  C9                      RET
                       ;++++
                       ;
```

```
                        ; CORCHK
                        ; CHECKS FREE MEMORY LIST FOR VALIDITY
                        ; CHAINS THRU THE LIST AND ATTEMPTS TO FIND A
                        ; POINTER BACK TO THE LISTHEAD WITHIN 256 ITERATIONS
                        ; IF NOT, ERROR MESSAGE PRINTED
                        ;RETURNS:
                        ;   SUCCEED:
                        ;       CARRY BIT CLEAR, SUCCEEDS IF LIST OK
                        ;   FAIL:
                        ;       CARRY BIT SET, FAILS IF LIST MESSED
                        ;
                        ;-----
0194'                   CORCHK:
0194'   E5                      PUSH    H               ;SAVE REGS
0195'   D5                      PUSH    D
0196'   C5                      PUSH    B
0197'   21 65E7                LXI     H,FWDPTR        ;HL->1ST ITEM
019A'   06FF                    MVI     B,0FFH          ;NO OF ITERATIONS
019C'   5E              ..C1:   MOV     E,M             ;DE -> NEXT ITEM
019D'   23                      INX     H
019E'   56                      MOV     D,M
019F'   EB                      XCHG                    ;HL -> NEXT ITEM
                        ;+
                        ; HERE WE CHECK THE CURRENT FORWARD POINTER AGAINST
                        ; THE LISTHEAD. IF HL = FWDPTR THEN WE ARE AT THE
                        ; END OF THE LIST AND CAN EXIT
                        ;-
01A0'   11 65E7                 LXI     D,FWDPTR
01A3'   7C                      MOV     A,H             ;CHECK HI ORDER
01A4'   BA                      CMP     D
01A5'   2004                    JRNZ    ..C2            ;NO? CONTINUE
01A7'   7D                      MOV     A,L             ;CHECK LO ORDER
01A8'   BB                      CMP     E
01A9'   2814                    JRZ     ..EX            ;YES? CAN EXIT
                        ;+
                        ; THE CURRENT ONE IS NOT THE LISTHEAD, IT'S SIZE
                        ; MUST BE BETWEEN 1 AND 255 (CANNOT BE 0)
                        ;-
01AB'   2B              ..C2:   DCX     H               ;HL->SIZE FIELD
                                TST     M               ;IS IT 0?[
01AC'   7E              +.IFN M         -A, [   MOV     A,M             ]
01AD'   B7              +               ORA     A]
01AE'   23                      INX     H               ;HL->PTR FIELD
01AF'   280B                    JRZ     ..ERR           ;YES? ERROR
                        ;+
                        ; THE CURRENT ADDRESS MUST ALSO BE BETWEEN THE
                        ; BOUNDS OF FREE MEMORY, RAMSTRT - RAMEND
                        ; IF IT IS NOT THEN IT IS AN INVALID ADDRESS
                        ;-
01B1'   7C                      MOV     A,H             ;A=HI PART OF ADDR
01B2'   FE69                    CPI     RAMSTRT>8       ;TOO SMALL?
01B4'   3806                    JRC     ..ERR           ;YES? ERROR
01B6'   FE80                    CPI     (RAMEND>8)+1    ;TOO BIG?
01B8'   3002                    JRNC    ..ERR           ;YES? ERROR!
01BA'   10E0                    DJNZ    ..C1            ;ITERATE AGAIN?
```

```
01BC'   37            ..ERR:  STC                     ;FAIL
01BD'   1801                  JMPR    ..OUT
01BF'                 ..EX:   CLC                     ;SUCCEEDI
01BF'   B7            +       ORA     A]
01C0'   C1            ..OUT:  POP     B
01C1'   D1                    POP     D
01C2'   E1                    POP     H
01C3'   C9                    RET
                              .END
```

ALLOC - ALLOC - ALLOCATION MODULE
+++++ SYMBOL TABLE +++++

AASN	005F		ALLOC	002E'	I	ALLOCD	0105'	I	ARGSTK	60CC	
BACKGR	65CC		BLANK	0020		BOTRAM	6000		BOTTOM	64A9	
CHAIN	000F'		CHARSL	65DD		CHKREC	007D'		CLEARS	60CA	
CNTRL	000C		CNTRLC	65CD		CNTRLO	65D9		CNTRLU	0015	
CNTRLZ	65E4		CORCHK	0194'	I	CORINI	014B'	I	CPLARE	611C	
CPLSIZ	0140		CR	000D		CSBLOK	668D		CSFLAG	6260	
CURCX	6814		CURCY	6816		CURREN	64A5		C.CO	0011	
C.C1	0012		C.CX	000B		C.CY	000D		C.DP	0013	
C.ST	0002		C.X	0003		C.XF	000F		C.XS	0007	
C.Y	0005		C.YF	0010		C.YS	0009		DDTON	65CE	
DECHAI	001E'		DEVBL	6589		DEVCL0	6579		DEVCL1	657B	
DEVCL2	657D		DEVCL3	657F		DEVCL4	6581		DEVCL5	6583	
DEVCL6	6585		DEVCL7	6587		DEVFB	65CB		DEVHCB	658F	
DEVMO	658B		DEVNM	65B7		DEVNT	658D		DEVTNA	65BB	
DEVTNB	65BF		DEVTNC	65C3		DEVVA	65BD		DEVVAR	6579	
DEVVB	65C1		DEVVBL	6591		DEVVC	65C5		DEVVD	65C9	
DEVVN	65B9		DEVVS	65C7		DEVXCD	65B3		DEVYCD	65B5	
DIV16	0000'	I	DOLPLH	62E8		DOLPPT	62EA		DUMBST	6577	
EDBCNT	64AD		EDCCNT	64A7		EDLONG	6812		EDMODE	681C	
EDNAME	64A1		EDNCX	680C		EDNCY	680E		EDNEWS	64A5	
EDOCX	6808		EDOCY	680A		EDPN	6806		EDPO	6804	
EDPTRC	64AB		EDPTRL	64A9		EDSTR	681A		ERABIT	0002	
ERRPGM	0000:04	X	ER.ARA	002F		ER.ARG	0034		ER.ASN	0015	
ER.BOX	001A		ER.CHN	0002		ER.CMD	001F		ER.CNV	0016	
ER.COR	001B		ER.CTL	0036		ER.DEL	0026		ER.DIM	0030	
ER.DIV	0018		ER.DP	0037		ER.DSK	0019		ER.EDT	0035	
ER.FMT	0038		ER.FNF	001C		ER.FOR	0028		ER.IMP	0003	
ER.LAB	0025		ER.MAC	0022		ER.NAE	002D		ER.NAM	0029	
ER.NEG	003A		ER.NOT	0023		ER.NUL	0039		ER.NUM	002B	
ER.NXT	001D		ER.OFL	0017		ER.OPN	0014		ER.OVE	001E	
ER.PAR	002A		ER.REN	002C		ER.RET	0024		ER.SEP	0021	
ER.SNP	0031		ER.SPC	002E		ER.STK	0004		ER.SW	0032	
ER.TER	0020		ER.UFL	0033		ER.UNF	0027		EXTDEL	002E	
E.HVAL	0002		E.LVAL	0001		E.SIZ	0005		E.TYP	0000	
E.VAL	0001		FCNTH	65D2		FCNTI	65D3		FCNTJ	65D4	
FCNTK	65D5		FCNTL	65D6		FCNTV	65E0		FCNTY	65E3	
FIRST	64A3		FLAGS	65CB		FOREGR	65D0		FRAGSI	0400	
FREE	00AD'	I	FREEAL	0133'	I	FREELS	65E5		FSTDOL	648C	
FSTINT	62EC		FWDPTR	65E7		HCAREA	6593		INCRO	65E9	
JUNK	6542		KBLOCK	65F1		KEYFLG	67FF		KEYPTK	6533	
KEYTRK	67F7		LF	000A		LISTON	65E2		MACSTU	6536	
MACTOP	625E		MAXFRG	0040		MNMX	65EB		NBLKB	0000	
NBLKM	0001		NEWBOT	6810		NL	000A		NSADDR	6802	
NUMBUF	64A3		N.LNK	0003		OLDCHR	64A0		OLDCUR	649F	
OLDKEY	649D		OLDXY	65EF		ONEBUF	6729		OPRL	0014	
OPRSP	655B		OPRSTK	6547		OPRSZ	655D		OUTOFF	62E7	
O.CHAR	0000		O.PREC	0002		O.SIZ	0006		O.SUB	0004	
O.TYP	0003		PCNT	655E		PIXVAL	65ED		POINTE	64A7	
PONOFF	65DA		PRINTR	6540		RAMEND	7FFF		RAMSTR	6900	
RANSHT	655F		RMDTMP	6538		RUBACK	0005		RUBOUT	007F	
SAVESP	625C		SCNEED	0004		SCRWIN	67CD		SOPRSP	653D	
SOPRSZ	653F		STACK	60C8		STAKTO	6000		STRSIZ	6800	
SUBSTU	6534		TAB	0009		TAPBUF	64B3		TAPCON	64AF	
TAPPRO	64B1		TBFEND	6533		TEMPHD	60CA		TEMPS	62E5	
TMPARG	67AD		TOP	6818		TTYBEG	6261		TTYEND	62E1	

ALLOC - ALLOC - ALLOCATION MODULE
+++++ SYMBOL TABLE +++++

TTYINT	62E3	TTYPTR	62E1	TXTWIN	67E2	UARTFL	649C
USREND	65F1	V3PTR	6573	VDCHAR	649E	VDNLF	65CF
VIPLH	6545	VOICE0	6563	WRMODE	65EE	ZGIM2	0001
ZGREND	681D	$AADR	0010	$ADDRF	0007	$ADDRI	0005
$ADDRS	0009	$ANY	001A	$ANYNA	FFFC	$ANYVA	FFFE
$ARGPT	0011	$BGPTR	000F	$BNDL	0007	$CALLE	000D
$CMDAD	0018	$CPLBL	002A	$CSBLO	0028	$DATAP	0007
$DOLDE	0001	$DOLOO	0002	$DVAL	0000	$END	001C
$FADR	000E	$FLAGS	0002	$FORBL	0024	$FORPT	000B
$FVAL	0006	$GOSUB	001A	$IADR	000C	$INPBU	0018
$INPPT	0016	$IVAL	0004	$KEYBL	0026	$LENGT	0001
$LINPT	0005	$LOCPT	0009	$MIBBL	0022	$MIBEN	001B
$NAMAD	000A	$NAME	000A	$NASCI	0009	$NDEL	0080
$NLINK	0003	$NULL	0002	$NVALU	0005	$REPEA	001E
$RVSTU	0013	$SAME	0020	$SASCI	000A	$SLEN	0006
$STRAD	0008	$STRPT	0003	$TAF	0000	$TYPE	0000
$USE	0005	.BLNK.	0000:03 X	.DATA.	0000* X	.PROG.	01C4 X

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                      ***
@                    *******
@                       *
@                      *
@                     *
@                    *
@                   *
@                  *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@               THE VERY BEST"
@
@             ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                  (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
         .LINK
         .PREL
         .IDENT  USEMAP
         .INTERN USEMAP
         .EXTERN ERRPGM
         .EXTERN FINDHASH
         .EXTERN NAMERR
         .EXTERN NLPNT
         .EXTERN NXTLNK
         .EXTERN NXTVAL
         .EXTERN PDECBC
         .EXTERN PRINTH
         .EXTERN PRTYPA
         .EXTERN TABPNT
;++++
;
; USEMAP
; COMMAND TO PRINT CORE DIRECTORY OF USED NAMES
; ON THE USERS TERMINAL
; FORMAT LOOKS LIKE:
; NAME  TYPE    SIZE
;   NAME - ASCII NAME OF THING
;   TYPE - TYPE OF THING (STRING, INTEGER)
;   SIZE - SIZE (IN BLOCKS) ALLOCATED TO IT
;
; CALLS:
;   FINDHASH - FIND HASH PTR FOR IT
;   NXTLNK - GET NEXT NAME BLOCK
;   DIRNAM - PRINT ENTRY FOR THE NAME
;
;----
         M.CMD[USEMAP,..USEX,0,AFIX[U],0,..USEMI[
0000'   18          +USEMAP: .BYTE $CMDADR
```

```
0001'   04              +       .BYTE (..USEX-USEMAP)/16+1
0002'   00              +       .BYTE 0
0003'   642C            +       .WORD AFIX[U][("U"-101Q)*16+FSTINT]
0005'   0011'          +       .WORD ..USEM
0007'   0000            +       .WORD 0
0009'   5553454D4150+   +       .ASCIZ /USEMAP/
                        +]
0010'   00                      .BYTE 0         ;NO SWITCHES
0011'                   ..USEM:
0011'   E5                      PUSH    H               ;SAVE PTR
0012'   21 62EC                 LXI     H,FSTINT        ;FIRST NAME HDR
0015'   3E40                    MVI     A,'A'-1         ;SECOND LETTER
0017'   F5                      PUSH    PSW             ;SAVED FOR NEXT
0018'   3E24                    MVI     A,'$'           ;THIS IS FIRST
001A'   CD 0000:05      ..U1:   CALL    FINDHASH        ;FIND HASH PTR
001D'   E5                      PUSH    H               ;SAVE IT
001E'   CD 0000:08      ..U2:   CALL    NXTLNK          ;GET NAME BLOCK
0021'   CA 002F'                JZ      ..U3            ;NO MORE HERE?
0024'   23                      INX     H               ; MOVE TO FLAGS
0025'   23                      INX     H
0026'   7E                      MOV     A,M
0027'   2B                      DCX     H
0028'   2B                      DCX     H
0029'   07                      RLC                     ; DO NOT PRINT
002A'   D4 003C'                CNC     DIRNAM          ; IF NO DELETE SET
002D'   18EF                    JMPR    ..U2            ;AND DO NEXT ONE
002F'   E1              ..U3:   POP     H               ;GET PTR
0030'   F1                      POP     PSW             ;AND LETTER
0031'   FE5A                    CPI     'Z'             ;WAS Z?
0033'   CA 003A'                JZ      ..USX           ;YES? EXIT THEN
0036'   3C                      INR     A               ;BUMP IT
0037'   F5                      PUSH    PSW             ;SAVE IT
0038'   18E0                    JMPR    ..U1            ;DIRC IT
003A'   E1              ..USX:  POP     H
003B'   C9                      RET
003C'                   ..USEX:
                        ;++++
                        ;
                        ; DIRNAM
                        ; PRINTS OUT A DIRCOR ENTRY FOR A GIVEN NAME
                        ;
                        ; NEEDS:
                        ;   HL -> NAME BLOCK
                        ;         **********
                        ;  HL-->* TYPE   * ($IADDR, $NAMADR)
                        ;         **********
                        ;         * SIZE   * (IN 16 BYTE BLOCKS)
                        ;         **********
                        ;         * NAME   *
                        ;         * PTR    *
                        ;         **********
                        ;         *  ...   *
                        ;         **********
                        ;
                        ; CALLS:
```

```
                        ;    PRINTHL - PRINT ASCII STRING
                        ;    PRTYPA - PRINT TYPE
                        ;    NXTVAL - GET VALUE FROM NAME
                        ;    TABPNT - PRINT TAB
                        ;    PDECBC - PRINT DECIMAL NUM
                        ;    NLPNT - PRINT NEWLINE
                        ;
                        ;-----
003C'                   DIRNAM:
003C'   E5                      PUSH    H
                        ;+
                        ; PRINT THE NAME
                        ;-
003D'   01 0009                 LXI     B,$NASCII       ;GET NAME FIELD
0040'   09                      DAD     B               ;HL->NAME
0041'   CD 0000:0B             CALL    PRINTHL         ;PRINT NAME
0044'   CD 0000:0D             CALL    TABPNT          ;AND TAB
                        ;+
                        ; PRINT THE TYPE
                        ;-
                        ; E IS THE SIZE OF THE NAME
0047'   AF                      XRA     A               ;CLEAR A
0048'   5F                      MOV     E,A             ;CLEAR E
0049'   E1                      POP     H               ;HL->TOP OF BLOCK
004A'   E5                      PUSH    H               ;SAVE AGAIN
004B'   7E                      MOV     A,M             ;A=TYPE
004C'   FE0A                    CPI     $NAMADR         ;IS NAME?
004E'   200A                    JRNZ    ..D1            ;NO? INT OR FLT
0050'   23                      INX     H               ;REMEMBER SIZE OF
0051'   5E                      MOV     E,M             ;NAME BLOCK FOR
0052'   2B                      DCX     H               ;SIZE LATER
0053'   CD 0000:09             CALL    NXTVAL          ;HL->VALUE
0056'   CA 0000:06             JZ      NAMERR          ;0? ERROR THEN
0059'   7E                      MOV     A,M             ;A=TYPE OF VAL
005A'   CD 0000:0C     ..D1:   CALL    PRTYPA          ;PRINT TYPE
005D'   CD 0000:0D             CALL    TABPNT          ;AND TAB
                        ;+
                        ; PRINT SIZE
                        ; E = SIZE OF NAME BLOCK IF NOT INT VALUE
                        ; ELSE E IS 0
                        ;-
0060'   23                      INX     H               ;HL->SIZE
0061'   7E                      MOV     A,M             ;A=SIZE OF VAL
0062'   83                      ADD     E               ;PLUS NAME SIZE
0063'   6F                      MOV     L,A             ;HL=SIZE IN BLOCKS
                        ;       CLR     H
0064'   AF                      XRA     A
0065'   67                      MOV     H,A             ;CLEAR H
0066'   29                      DAD     H               ;HL=SIZE IN BYTES
0067'   29                      DAD     H
0068'   29                      DAD     H
0069'   29                      DAD     H
                        ;       MVD     B,H             ;BC=SIZE TO PRINT
006A'   E5                      PUSH    H
006B'   C1                      POP     B
```

```
006C'   CD 0000:0A          CALL    PDECBC          ;PRINT SIZE
006F'   CD 0000:07          CALL    NLPNT           ;AND NEWLINE
0072'   E1                  POP     H               ;RESTORE REGS
0073'   C9                  RET
                            .END
```

USEMAP -
+++++ SYMBOL TABLE +++++

AASN	005F	ARGSTK	60CC	BACKGR	65CC	BLANK	0020
BOTRAM	6000	BOTTOM	64A9	CHARSL	65DD	CLEARS	60CA
CNTRL	000C	CNTRLC	65CD	CNTRLO	65D9	CNTRLU	0015
CNTRLZ	65E4	CPLARE	611C	CPLSIZ	0140	CR	000D
CSBLOK	668D	CSFLAG	6260	CURCX	6814	CURCY	6816
CURREN	64A5	C.C0	0011	C.C1	0012	C.CX	000B
C.CY	000D	C.DP	0013	C.ST	0002	C.X	0003
C.XF	000F	C.XS	0007	C.Y	0005	C.YF	0010
C.YS	0009	DDTON	65CE	DEVBL	6589	DEVCL0	6579
DEVCL1	657B	DEVCL2	657D	DEVCL3	657F	DEVCL4	6581
DEVCL5	6583	DEVCL6	6585	DEVCL7	6587	DEVFB	65CB
DEVHCB	658F	DEVMO	658B	DEVNM	65B7	DEVNT	658D
DEVTNA	65BB	DEVTNB	65BF	DEVTNC	65C3	DEVVA	65BD
DEVVAR	6579	DEVVB	65C1	DEVVBL	6591	DEVVC	65C5
DEVVD	65C9	DEVVN	65B9	DEVVS	65C7	DEVXCD	65B3
DEVYCD	65B5	DIRNAM	003C	DOLPLH	62E8	DOLPPT	62EA
DUMBST	6577	EDBCNT	64AD	EDCCNT	64A7	EDLONG	6812
EDMODE	681C	EDNAME	64A1	EDNCX	680C	EDNCY	680E
EDNEWS	64A5	EDOCX	6808	EDOCY	680A	EDPN	6806
EDPO	6804	EDPTRC	64AB	EDPTRL	64A9	EDSTR	681A
ERABIT	0002	ERRPGM	0000:04 X	ER.ARA	002F	ER.ARG	0034
ER.ASN	0015	ER.BOX	001A	ER.CHN	0002	ER.CMD	001F
ER.CNV	0016	ER.COR	001B	ER.CTL	0036	ER.DEL	0026
ER.DIM	0030	ER.DIV	0018	ER.DP	0037	ER.DSK	0019
ER.EDT	0035	ER.FMT	0038	ER.FNF	001C	ER.FOR	0028
ER.IMP	0003	ER.LAB	0025	ER.MAC	0022	ER.NAE	002D
ER.NAM	0029	ER.NEG	003A	ER.NOT	0023	ER.NUL	0039
ER.NUM	002B	ER.NXT	001D	ER.OFL	0017	ER.OPN	0014
ER.OVE	001E	ER.PAR	002A	ER.REN	002C	ER.RET	0024
ER.SEP	0021	ER.SNP	0031	ER.SPC	002E	ER.STK	0004
ER.SW	0032	ER.TER	0020	ER.UFL	0033	ER.UNF	0027
EXTDEL	002E	E.HVAL	0002	E.LVAL	0001	E.SIZ	0005
E.TYP	0000	E.VAL	0001	FCNTH	65D2	FCNTI	65D3
FCNTJ	65D4	FCNTK	65D5	FCNTL	65D6	FCNTV	65E0
FCNTY	65E3	FINDHA	0000:05 X	FIRST	64A3	FLAGS	65CB
FOREGR	65D0	FRAGSI	0400	FREELS	65E5	FSTDOL	648C
FSTINT	62EC	FWDPTR	65E7	HCAREA	6593	INCRO	65E9
JUNK	6542	KBLOCK	65F1	KEYFLG	67FF	KEYPTK	6533
KEYTRK	67F7	LF	000A	LISTON	65E2	MACSTU	6536
MACTOP	625E	MAXFRG	0040	MNMX	65EB	NAMERR	0000:06 X
NBLKB	0000	NBLKM	0001	NEWBOT	6810	NL	000A
NLPNT	0000:07 X	NSADDR	6802	NUMBUF	64A3	NXTLNK	0000:08 X
NXTVAL	0000:09 X	OLDCHR	64A0	OLDCUR	649F	OLDKEY	649D
OLDXY	65EF	ONEBUF	6729	OPRL	0014	OPRSP	655B
OPRSTK	6547	OPRSZ	655D	OUTOFF	62E7	PCNT	655E
PDECBC	0000:0A X	PIXVAL	65ED	POINTE	64A7	PONOFF	65DA
PRINTH	0000:0B X	PRINTR	6540	PRTYPA	0000:0C X	RAMEND	7FFF
RAMSTR	6900	RANSHT	655F	RMDTMP	6538	RUBACK	0005
RUBOUT	007F	SAVESP	625C	SCNEED	0004	SCRWIN	67CD
SOPRSP	653D	SOPRSZ	653F	STACK	60C8	STAKTO	6000
STRSIZ	6800	SUBSTU	6534	TAB	0009	TABPNT	0000:0D X
TAPBUF	64B3	TAPCON	64AF	TAPPRO	64B1	TBFEND	6533
TEMPHD	60CA	TEMPS	62E5	TMPARG	67AD	TOP	6818
TTYBEG	6261	TTYEND	62E1	TTYINT	62E3	TTYPTR	62E1
TXTWIN	67E2	UARTFL	649C	USEMAP	0000 I	USREND	65F1

USEMAP -
+++++ SYMBOL TABLE +++++

```
V3PTR   6573        VDCHAR  649E        VDNLF   65CF        VIPLH   6545
VOICE0  6563        WRMODE  65EE        ZGIM2   0001        ZGREND  681D
$AADR   0010        $ADDRF  0007        $ADDRI  0005        $ADDRS  0009
$ANY    001A        $ANYNA  FFFC        $ANYVA  FFFE        $ARGPT  0011
$BGPTR  000F        $BNDL   0007        $CALLE  000D        $CMDAD  0018
$CPLBL  002A        $CSBLO  0028        $DATAP  0007        $DOLDE  0001
$DOLOD  0002        $DVAL   0000        $END    001C        $FADR   000E
$FLAGS  0002        $FORBL  0024        $FORPT  000B        $FVAL   0006
$GOSUB  001A        $IADR   000C        $INPBU  0018        $INPPT  0016
$IVAL   0004        $KEYBL  0026        $LENGT  0001        $LINPT  0005
$LOCPT  0009        $MIBBL  0022        $MIBEN  001B        $NAMAD  000A
$NAME   000A        $NASCI  0009        $NDEL   0080        $NLINK  0003
$NULL   0002        $NVALU  0005        $REPEA  001E        $RVSTU  0013
$SAME   0020        $SASCI  000A        $SLEN   0006        $STRAD  0008
$STRPT  0003        $TAF    0000        $TYPE   0000        $USE    0005
.BLNK.  0000:03 X   .DATA.  0000*   X   .PROG.  0074'   X
```

```
 .INSERT A:S.ASM
@.REMARK /
@                        *
@                      * * *
@                       ***
@                     *******
@                        *
@                       *
@                      *
@                     *
@                    *
@                   *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                THE VERY BEST"
@
@              ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                   (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
 .INSERT A:MAC.ASM
@
 .INSERT A:NCUEQU.ASM
         .LINK
         .PREL
         .IDENT  MOD
         .INTERN MODIFY
         .EXTERN ARG
         .EXTERN CARTYP
         .EXTERN DONCU
         .EXTERN ERRPGM
         .EXTERN HEXHL
         .EXTERN HEXBYT
         .EXTERN IGET
         .EXTERN INHEX
         .EXTERN INPTTY
         .EXTERN IPOP
         .EXTERN IPUSH
         .EXTERN IYNCU
         .EXTERN KLB
         .EXTERN NAMADR
         .EXTERN NAMSET
         .EXTERN NLPNT
         .EXTERN NXTABC
         .EXTERN NXTLNK
         .EXTERN NXTPTR
         .EXTERN NXTVAL
         .EXTERN OUTCH
         .EXTERN PDECBC
         .EXTERN PRINTF
         .EXTERN PRINTHL
         .EXTERN PRTYPA
         .EXTERN SRCHOPR
```

```
                                .EXTERN TABPNT

                        ;MODIFY COMMAND
                        M.CMD[MODIFY,..MODX,0,MF,..MODA,..MOD1]C
0000'   18              +MODIFY: .BYTE $CMDADR
0001'   0D              +        .BYTE (..MODX-MODIFY)/16+1
0002'   00              +        .BYTE 0
0003'   00C1'           +        .WORD MF
0005'   0030'           +        .WORD ..MOD1
0007'   0010'           +        .WORD ..MODA
0009'   4D4F44494659+           .ASCIZ /MODIFY/
                        +]
0010'   63               ..MODA: .BYTE 99          ;SPECIAL
0011'   484558455850             .ASCIZ /HEXEXPR (I.E. NAME+HEX OR HEX)/
0030'   CD 0000:10      ..MOD1: CALL KLB          ;KILL BLANX
0033'   DDE5                    PUSH X            ;SAVE X
0035'   CD 0059'                CALL ..GHEX       ;GET HEX
0038'   D5                      PUSH D            ;SAVE VALUE
0039'   CD 0074'        ..MORE: CALL ..OPR        ;GET OPERATOR
003C'   3850                    JRC ..EX
003E'   CD 0059'                CALL ..GHEX       ;NEXT VALUE
0041'   D5                      PUSH D
0042'   E5                      PUSH    H         ;GET THE SUBROUTINE
0043'   DD6E04                  MOV     L,0.SUB(X)
0046'   DD6605                  MOV     H,0.SUB+1(X)
0049'   E5                      PUSH    H
004A'   DDE1                    POP     X         ;INTO X
004C'   E1                      POP     H
                                JSR X[
004D'   DDE5            +.IFIDN [X] [X], [        PUSH    X
004F'   DD21 0057'      +       LXI     X,..0001
0053'   DDE3           +       XTIX
0055'   DDE9           +       PCIX
                        +]
0057'   18E0                    JMPR ..MORE       ;GOON
0059'   CD 0000:05       ..GHEX: CALL CARTYP       ;IS ALPHA, ETC.
                                ZERROR ER.NUM[
005C'   2004           +       JRNZ ..0002
005E'   CD 0000:07      +       CALL ERRPGM
0061'   2B              +       .BYTE ER.NUM
0062'                   +..0002:]
0062'   3004                    JRNC ..ALPH
0064'   CD 0000:0B              CALL INHEX
0067'   C9                      RET
0068'   CD 0000:12       ..ALPH: CALL NAMSET
006B'   AF                      XRA A
006C'   CD 0000:11              CALL NAMADR
006F'   C8                      RZ
                                ERROR ER.NAME[
0070'   CD 0000:07      +       CALL ERRPGM
0073'   29              +       .BYTE ER.NAME
                        +]
0074'   C5               ..OPR: PUSH B            ;SEE NOLA'S EVAL
0075'   D5                      PUSH D
0076'   4E                      MOV C,M
```

```
0077'   23                      INX H
0078'   11 00B4'                LXI D,..MTAB
007B'   C3 0000:1D              JMP SRCHOPR
007E'   C1              ..MADD: POP B
007F'   D1                      POP D
0080'   E3                      XTHL
0081'   19                      DAD D
0082'   E3                      XTHL
0083'   C5                      PUSH B
0084'   C9                      RET
0085'   C1              ..MSUB: POP B
0086'   D1                      POP D
0087'   E3                      XTHL
                                CLC[
0088'   B7              +       ORA     A]
0089'   ED52                    DSBC D
008B'   E3                      XTHL
008C'   C5                      PUSH B
008D'   C9                      RET
008E'   D1              ..EX:   POP D           ;GET VALUE OFF STACK
008F'   CD 0000:13      ..DEX:  CALL NLPNT
0092'   7A                      MOV A,D         ;PRINT ADDRESS STRAIGHT
0093'   CD 0000:09              CALL HEXBYT
0096'   7B                      MOV A,E
0097'   CD 0000:09              CALL HEXBYT
009A'   3E3D                    MVI A,'='
009C'   CD 0000:18              CALL OUTCH
009F'   1A                      LDAX D          ;GET VALUE BACKWARDS
00A0'   CD 0000:09              CALL HEXBYT
00A3'   13                      INX D
00A4'   1A                      LDAX D
00A5'   CD 0000:09              CALL HEXBYT
00A8'   13                      INX D
00A9'   7E                      MOV A,M
00AA'   FE2C                    CPI ',          ;COMMA MEANS PRINT ON
00AC'   28E1                    JRZ ..DEX
00AE'   CD 0000:13              CALL NLPNT
00B1'   DDE1                    POP X
00B3'   C9                      RET
00B4'                   ..MTAB: OPER '+',,0,0,..MADD[
00B4'   2B              +       .BYTE   '+'
00B5'   00              +.IFB [], [.BYTE 0] [.BYTE ]
00B6'   00              +       .BYTE   0
00B7'   00              +       .BYTE   0
00B8'   007E'           +       .WORD   ..MADD]
                                OPER '-',,0,0,..MSUB[
00BA'   2D              +       .BYTE   '-'
00BB'   00              +.IFB [], [.BYTE 0] [.BYTE ]
00BC'   00              +       .BYTE   0
00BD'   00              +       .BYTE   0
00BE'   0085'           +       .WORD   ..MSUB]
00C0'   00                      .BYTE 0
00C1'                   ..MODX:
                        ; MF COMMAND - PRINTS OUT FLOATING POINT VALUE OF RATION
                        AL NUMBER
```

MOD -

```
                        ; A/B, COMMAND FORMAT: MF A,B
                        M.CMD[MF,..MFX,0,AFIX[M],..MFA,..MFS][
00C1'   18              +MF:        .BYTE $CMDADR
00C2'   04              +           .BYTE (..MFX-MF)/16+1
00C3'   00              +           .BYTE 0
00C4'   63AC            +           .WORD AFIX[M][("M"-1010)*16+FSTINT]
00C6'   00CD'           +           .WORD ..MFS
00C8'   00D0'           +           .WORD ..MFA
00CA'   4D4600          +           .ASCIZ /MF/
                        +]
00CD'   CD 0000:04      ..MFS:      CALL    ARG
00D0'   0600            ..MFA:      .BYTE   $FVAL,$TAF
00D2'   E5                          PUSH    H
00D3'   CD 0000:0F                  CALL    IYNCU
00D6'   CD 0000:0D                  CALL    IPOP
00D9'   7A                          MOV     A,D
00DA'   CD 0000:09                  CALL    HEXBYT
00DD'   7B                          MOV     A,E
00DE'   CD 0000:09                  CALL    HEXBYT
00E1'   CD 0000:0D                  CALL    IPOP
00E4'   7A                          MOV     A,D
00E5'   CD 0000:09                  CALL    HEXBYT
00E8'   7B                          MOV     A,E
00E9'   CD 0000:09                  CALL    HEXBYT
00EC'   CD 0000:13                  CALL    NLPNT
00EF'   E1                          POP     H
00F0'   C9                          RET
00F1'                   ..MFX:

                        .END
```

MOD -
+++++ SYMBOL TABLE +++++

AASN	005F	ARG	0000:04 X	ARGSTK	60CC	BACKGR	65CC
BLANK	0020	BOTRAM	6000	BOTTOM	64A9	CARTYP	0000:05 X
CHARSL	65DD	CLEARS	60CA	CNTRL	000C	CNTRLC	65CD
CNTRLO	65D9	CNTRLU	0015	CNTRLZ	65E4	CPLARE	611C
CPLSIZ	0140	CR	000D	CSBLOK	668D	CSFLAG	6260
CURCX	6814	CURCY	6816	CURREN	64A5	C.C0	0011
C.C1	0012	C.CX	000B	C.CY	000D	C.DP	0013
C.ST	0002	C.X	0003	C.XF	000F	C.XS	0007
C.Y	0005	C.YF	0010	C.YS	0009	DDTON	65CE
DEVBL	6589	DEVCL0	6579	DEVCL1	657B	DEVCL2	657D
DEVCL3	657F	DEVCL4	6581	DEVCL5	6583	DEVCL6	6585
DEVCL7	6587	DEVFB	65CB	DEVHCB	658F	DEVMO	658B
DEVNM	65B7	DEVNT	658D	DEVTNA	65BB	DEVTNB	65BF
DEVTNC	65C3	DEVVA	65BD	DEVVAR	6579	DEVVB	65C1
DEVVBL	6591	DEVVC	65C5	DEVVD	65C9	DEVVN	65B9
DEVVS	65C7	DEVXCD	65B3	DEVYCD	65B5	DOLPLH	62E8
DOLPPT	62EA	DONCU	0000:06 X	DUMBST	6577	EDBCNT	64AD
EDCCNT	64A7	EDLONG	6812	EDMODE	681C	EDNAME	64A1
EDNCX	680C	EDNCY	680E	EDNEWS	64A5	EDOCX	6808
EDOCY	680A	EDPN	6806	EDPO	6804	EDPTRC	64AB
EDPTRL	64A9	EDSTR	681A	ERABIT	0002	ERRPGM	0000:07 X
ER.ARA	002F	ER.ARG	0034	ER.ASN	0015	ER.BOX	001A
ER.CHN	0002	ER.CMD	001F	ER.CNV	0016	ER.COR	001B
ER.CTL	0036	ER.DEL	0026	ER.DIM	0030	ER.DIV	0018
ER.DP	0037	ER.DSK	0019	ER.EDT	0035	ER.FMT	0038
ER.FNF	001C	ER.FOR	0028	ER.IMP	0003	ER.LAB	0025
ER.MAC	0022	ER.NAE	002D	ER.NAM	0029	ER.NEG	003A
ER.NOT	0023	ER.NUL	0039	ER.NUM	002B	ER.NXT	001D
ER.OFL	0017	ER.OPN	0014	ER.OVE	001E	ER.PAR	002A
ER.REN	002C	ER.RET	0024	ER.SEP	0021	ER.SNP	0031
ER.SPC	002E	ER.STK	0004	ER.SW	0032	ER.TER	0020
ER.UFL	0033	ER.UNF	0027	EXTDEL	002E	E.HVAL	0002
E.LVAL	0001	E.SIZ	0005	E.TYP	0000	E.VAL	0001
FCNTH	65D2	FCNTI	65D3	FCNTJ	65D4	FCNTK	65D5
FCNTL	65D6	FCNTV	65E0	FCNTY	65E3	FIRST	64A3
FLAGS	65CB	FOREGR	65D0	FRAGSI	0400	FREELS	65E5
FSTDOL	648C	FSTINT	62EC	FWDPTR	65E7	HCAREA	6593
HEXBYT	0000:09 X	HEXHL	0000:08 X	IGET	0000:0A X	INCRO	65E9
INHEX	0000:0B X	INPTTY	0000:0C X	IPOP	0000:0D X	IPUSH	0000:0E X
IYNCU	0000:0F X	JUNK	6542	KBLOCK	65F1	KEYFLG	67FF
KEYPTK	6533	KEYTRK	67F7	KLB	0000:10 X	LF	000A
LISTON	65E2	MACSTU	6536	MACTOP	625E	MAXFRG	0040
MF	00C1	MNMX	65EB	MODIFY	0000 I	M.POPS	0078
NAMADR	0000:11 X	NAMSET	0000:12 X	NBLKB	0000	NBLKM	0001
NCUCOM	0029	NCUDAT	0028	NEWBOT	6810	NL	000A
NLPNT	0000:13 X	NSADDR	6802	NUMBUF	64A3	NXTABC	0000:14 X
NXTLNK	0000:15 X	NXTPTR	0000:16 X	NXTVAL	0000:17 X	N.ACOS	0006
N.ARG	0003	N.ASIN	0005	N.ATAN	0007	N.CHSD	0034
N.CHSF	0015	N.CHSS	0074	N.COS	0003	N.DADD	002C
N.DDIV	002F	N.DIV	0004	N.DMUL	002E	N.DMUU	0036
N.DSUB	002D	N.EMAX	0017	N.EMIN	FF69	N.ESGN	0040
N.EXP	000A	N.FADD	0010	N.FDIV	0013	N.FIXD	001E
N.FIXS	001F	N.FLTD	001C	N.FLTS	001D	N.FMUL	0012
N.FSUB	0011	N.LN	0009	N.LOG	0008	N.MSGN	0080
N.NOP	0000	N.OFLO	0001	N.POPD	0038	N.POPF	0018

MOD -
+++++ SYMBOL TABLE +++++

N.PTOD	0037	N.PTOF	0017	N.PTOS	0077	N.PUPI	001A
N.PWR	000B	N.SADD	006C	N.SDIV	006F	N.SIGN	0040
N.SIN	0002	N.SMUL	006E	N.SMUU	0076	N.SQRT	0001
N.SSUB	006D	N.TAN	0004	N.UFLO	0002	N.XCHD	0039
N.XCHF	0019	N.XCHS	0079	N.ZERO	0020	OLDCHR	64A0
OLDCUR	649F	OLDKEY	649D	OLDXY	65EF	ONEBUF	6729
OPRL	0014	OPRSP	655B	OPRSTK	6547	OPRSZ	655D
OUTCH	0000:18 X	OUTOFF	62E7	O.CHAR	0000	O.PREC	0002
O.SIZ	0006	O.SUB	0004	O.TYP	0003	PCNT	655E
PDECBC	0000:19 X	PIXVAL	65ED	POINTE	64A7	PONOFF	65DA
PRINTF	0000:1A X	PRINTH	0000:1B X	PRINTR	6540	PRTYPA	0000:1C X
RAMEND	7FFF	RAMSTR	6900	RANSHT	655F	RMDTMP	6538
RUBACK	0005	RUBOUT	007F	SAVESP	625C	SCNEED	0004
SCRWIN	67CD	SOPRSP	653D	SOPRSZ	653F	SRCHOP	0000:1D X
STACK	60C8	STAKTO	6000	STRSIZ	6800	SUBSTU	6534
TAB	0009	TABPNT	0000:1E X	TAPBUF	64B3	TAPCON	64AF
TAPPRO	64B1	TBFEND	6533	TEMPHD	60CA	TEMPS	62E5
TMPARG	67AD	TOP	6818	TTYBEG	6261	TTYEND	62E1
TTYINT	62E3	TTYPTR	62E1	TXTWIN	67E2	UARTFL	649C
USREND	65F1	V3PTR	6573	VDCHAR	649E	VDNLF	65CF
VIPLH	6545	VOICEO	6563	WRMODE	65EE	ZGIM2	0001
ZGREND	681D	$AADR	0010	$ADDRF	0007	$ADDRI	0005
$ADDRS	0009	$ANY	001A	$ANYNA	FFFC	$ANYVA	FFFE
$ARGPT	0011	$BGPTR	000F	$BNDL	0007	$CALLE	000D
$CMDAD	0018	$CPLBL	002A	$CSBLO	0028	$DATAP	0007
$DOLDE	0001	$DOLOO	0002	$DVAL	0000	$END	001C
$FADR	000E	$FLAGS	0002	$FORBL	0024	$FORPT	000B
$FVAL	0006	$GOSUB	001A	$IADR	000C	$INPBU	0018
$INPPT	0016	$IVAL	0004	$KEYBL	0026	$LENGT	0001
$LINPT	0005	$LOCPT	0009	$MIBBL	0022	$MIBEN	001B
$NAMAD	000A	$NAME	000A	$NASCI	0009	$NDEL	0080
$NLINK	0003	$NULL	0002	$NVALU	0005	$REPEA	001E
$RVSTU	0013	$SAME	0020	$SASCI	000A	$SLEN	0006
$STRAD	0008	$STRPT	0003	$TAF	0000	$TYPE	0000
$USE	0005	.BLNK.	0000:03 X	.DATA.	0000* X	.PROG.	00F1' X

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                      ***
@                     *******
@                         *
@                        *
@                       *
@                      *
@                     *
@                     *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                THE VERY BEST"
@
@              ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                    (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
          .LINK
          .PREL
          .IDENT  SUB
          .EXTERN ALLOC
          .EXTERN ARGLIN
          .EXTERN DELMIB
          .EXTERN ERRPGM
          .EXTERN FREE
          .EXTERN GETOPND
          .EXTERN OUTCH
          .INTERN ACOPY
          .INTERN CARTYA
          .INTERN CARTYP
          .INTERN CLEARI
          .INTERN DEC5IY
          .INTERN FINDLAB
          .INTERN GETLPT
          .INTERN HEXBYT
          .INTERN HEXHL
          .INTERN INC5IY
          .INTERN INHEX
          .INTERN ISNL
          .INTERN ISTERM
          .INTERN KISNL
          .INTERN KLB
          .INTERN NLPNT
          .INTERN NXTABC
          .INTERN NXTLNK
          .INTERN NXTPTR
          .INTERN NXTVAL
          .INTERN PDECA
          .INTERN PDECBC
          .INTERN PRINTA
```

```
                                .INTERN PRINTFF
                                .INTERN PRINTHL
                                .INTERN PUTLPT
                                .INTERN SCANB
                                .INTERN SCANLAB
                                .INTERN SCANNL
                                .INTERN STREQK
                                .INTERN SWITCH
                                .INTERN TABPNT
                                .INTERN TEMPCHN
                                .INTERN TEMPUNCH
                                .INTERN WARNFF

0000'   CD 000D'        WARNFF: CALL PRINTFF
0003'   5741524E494E            .ASCIZ /WARNING: /

                        ;PRINTFF--PRINT STRING FOLLOWING IN LINE TILL
                        ;NULL IS SEEN

000D'   E3              PRINTFF: XTHL           ;SAVE HL, GET PTR
000E'   7E              ..LOOP: MOV A,M         ;GET THE CHARACTER
000F'   23                      INX H           ;BUMP HL
0010'   A7                      ANA A           ;IS NULL?
0011'   2805                    JRZ ..QUIT      ;YUP
0013'   CD 0000:0A             CALL OUTCH      ;PUTITOUT
0016'   18F6                    JMPR ..LOOP     ;MORE
0018'   E3              ..QUIT: XTHL            ;AGAIN
0019'   C9                      RET

                        ;PRINTHL--PRINT STRING HL->AT UNTIL A NULL

001A'   AF              PRINTHL: XRA A  ;NULL A

                        ;PRINTA--PRINTS WHAT HL->AT UNTIL DELIM
                        ;         IN A SEEN OR A NULL IS FOUND
                        ;HL IS ADVANCED TO POINT PAST DELIMITER

001B'   C5              PRINTA: PUSH B
001C'   4F                      MOV C,A         ;COPY
001D'   7E              ..LOOP: MOV A,M         ;GET CHAR
001E'   23                      INX H
001F'   B7                      ORA A           ;IS NULL?
0020'   2808                    JRZ ..QUIT      ;YAS
0022'   B9                      CMP C
0023'   2805                    JRZ ..QUIT
0025'   CD 0000:0A             CALL OUTCH
0028'   18F3                    JMPR ..LOOP
002A'   C1              ..QUIT: POP B
002B'   C9                      RET

                        ;NLPNT AND TABPNT--PRINTS NL'S AND TABS

002C'   3E0A            NLPNT:  MVI A,NL
002E'   1802                    JMPR TABPN1
0030'   3E20            TABPNT: MVI A,' '
```

```
0032'  CD 0000:0A   TABPN1: CALL OUTCH
0035'  C9                   RET
                    ;PDECA--PRINTS CONTENTS OF A AS UNSIGNED
                    ;         DECIMAL NUMBER

0036'  C5           PDECA:  PUSH B  ;SAVE BC
0037'  4F                   MOV C,A ;STORE A IN BC
0038'  0600                 MVI B,0
003A'  1801                 JMPR PDECB1     ;PRINT BC

                    ;PDECBC--PRINT BC AS 16-BIT SIGNED #

003C'  C5           PDECBC: PUSH B
003D'  D5           PDECB1: PUSH D
003E'  E5                   PUSH H
003F'  21 0082'             LXI H,..DTBL    ;HL->DIVIDE TABLE
0042'  E5                   PUSH H
0043'  60                   MOV H,B         ;PUT BC IN HL
0044'  69                   MOV L,C
0045'  78                   MOV A,B         ;CHECK FOR NEGATIVE
0046'  A7                   ANA A
0047'  F2 0052'             JP ..PLUS
004A'  3E2D                 MVI A,'-'       ;PUT OUT A -
004C'  CD 0000:0A           CALL OUTCH
004F'  CD 008C'             CALL ABS        ;;NEGATE IT
0052'                ..PLUS:
                    ;NOW WE HAVE HL=WHAT'S LEFT, SP->DIV TABLE
                    ;USE B FOR LOOP COUNT AND C=LZ SUPPRESS FLAG
                    ;DE IS THE SUBTRACTOR
0052'  01 0501              LXI B,501H      ;INITIALIZE (SURE)
0055'  E3           ..LOOP: XTHL            ;HL->TOP
0056'  5E                   MOV E,M         ;GET SUBRTRACTOR
0057'  23                   INX H
0058'  56                   MOV D,M
0059'  23                   INX H
005A'  E3                   XTHL            ;BACK TO WHAT'S LEFT
005B'  AF                   XRA A           ;CLEAR CURRENT DIGIT
005C'  ED52          ..SUBT: DSBC D
005E'  CB7C                 BIT 7,H ;DID WE GO -?
0060'  2003                 JRNZ ..MINU
0062'  3C                   INR A
0063'  18F7                 JMPR ..SUBT
                    ;RESULT WENT NEGATIVE, CORRECT IT
0065'  19           ..MINU: DAD D
0066'  A7                   ANA A           ;IS DIGIT 0?
0067'  2003                 JRNZ ..PRNT
0069'  B1                   ORA C           ;HAVE WE PRINTED A 0
                                             ;BEFORE?
006A'  2007                 JRNZ ..SKLZ
006C'  C630          ..PRNT: ADI '0'         ;MAKE INTO ASCII
006E'  CD 0000:0A           CALL OUTCH      ;AND PRINT IT
0071'  0E00                 MVI C,0         ;SET LZ FLAG
0073'  10E0          ..SKLZ: DJNZ ..LOOP     ;LOOP BOTTOM
0075'  0D                   DCR C           ;WAS NUMBER ZERO?
0076'  2005                 JRNZ ..DONE     ;NOPE
```

```
0078'  3E30              MVI A,'0'       ;PUT OUT A ZERO
007A'  CD 0000:0A        CALL OUTCH
007D'  E1          ..DONE: POP H
007E'  E1                  POP H
007F'  D1                  POP D
0080'  C1                  POP B
0081'  C9                  RET
                   ;DIVIDE TABLE FOLLOWS
0082'  2710  03E8   DTBL: .WORD 10000,1000,100,10,1


                   ;COMPUTE ABS (HL)
008C'  7C          ABS:    MOV A,H
008D'  A7                  ANA A
008E'  F0                  RP
008F'  2F                  CMA
0090'  67                  MOV H,A
0091'  7D                  MOV A,L
0092'  2F                  CMA
0093'  6F                  MOV L,A
0094'  23                  INX H
0095'  C9                  RET

                   ;INHEX--GETS VALUE IN DE FROM HEX CHARS HL->AT

0096'  C5          INHEX:  PUSH B
0097'  11 0000             LXI D,0         ;ACCUMULATOR
009A'  CD 025F'    ..SCAN: CALL CARTYP     ;IS NUMBER OR NAME?
009D'  7E                  MOV A,M         ;GET CHAR ANYWAY
009E'  2815                JRZ ..INDL      ;IS DELIM
00A0'  3004                JRNC ..ALPH     ;IS NAME
00A2'  D630                SUI 30H         ;NUMMIE
00A4'  1802                JMPR ..ADD      ;ADD IT IN
00A6'  D637        ..ALPH: SUI 37H         ;SHOULD BE A-F
00A8'  EB          ..ADD:  XCHG
00A9'  4F                  MOV C,A
00AA'  0600                MVI B,0
00AC'  29                  DAD H
00AD'  29                  DAD H
00AE'  29                  DAD H
00AF'  29                  DAD H   ;MUL BY 16
00B0'  09                  DAD B   ;AND ADD IN E#NEW ONE
00B1'  EB                  XCHG    ;PUT IN D
00B2'  23                  INX H   ;NEXT CHAR
00B3'  18E5                JMPR ..SCAN
00B5'  C1          ..INDL: POP B
00B6'  C9                  RET

                   ;HEXHL--PRINTS BYTE HL->AT IN HEX

00B7'  7E          HEXHL:  MOV A,M
00B8'  F5          HEXBYT: PUSH PSW
00B9'  0F                  RRC
00BA'  0F                  RRC
00BB'  0F                  RRC
```

```
00BC'   0F                      RRC
00BD'   CD 00C1'                CALL ..HNIB
00C0'   F1                      POP PSW
00C1'   E60F            ..HNIB: ANI 0FH
00C3'   FE0A                    CPI 10

00C5'   3802                    JRC ..LT10      ;<10
00C7'   C607                    ADI 'A'-'9'-1   ;ADD OFFSET
00C9'   C630            ..LT10: ADI '0'         ;MAKE INTO ASCII
00CB'   CD 0000:0A              CALL OUTCH
00CE'   C9                      RET

                        ; *TEMPCHN--LINKS HL INTO TEMP LIST
                        ;IN:    HL->BLOCK TO BE LINKED IN
                        ;OUT:   SAME
                        ;
                        ;NOTE:  BLOCKS LINKED IN SHOULD NOT HAVE STUFF
                        ;       HANGING OFF, THAT IS, NAMEBLOX WITH
                        ;       POINTER VALUES SHOULDN'T BE TEMPCHNED

00CF'                   TEMPCHN:
00CF'   E5                      PUSH H
00D0'   D5                      PUSH D
00D1'   C5                      PUSH B
00D2'   ED5B 60CA               LDED TEMPHDR    ;PICKUP CURRENT FIRST
00D6'   22 60CA                 SHLD TEMPHDR    ;STUFF NEW GUY
00D9'   01 0003                 LXI B,$NLINK
00DC'   09                      DAD B
00DD'   73                      MOV M,E
00DE'   23                      INX H
00DF'   72                      MOV M,D
00E0'   C1                      POP B
00E1'   D1                      POP D
00E2'   E1                      POP H
00E3'   C9                      RET

                        ; *TEMPUNCH--PULLS HL OUT OF TEMP LIST
                        ;IN:    HL->BLOCK TO BE EXTRACTED
                        ;OUT:   BLOCK EXTRACTED

00E4'                   TEMPUNCH:
00E4'   C5                      PUSH B
00E5'   DDE5                    PUSH X
00E7'   E5                      PUSH H
00E8'   21 60C7                 LXI H,TEMPHDR-$NLINK    ;LISTHEAD
00EB'   E5              ..MORE: PUSH H          ;SAVE POINTER IN X
00EC'   DDE1                    POP X
00EE'   CD 0227                 CALL NXTLNK     ;GET NEX GUY
00F1'   2005                    JRNZ ..THERE    ;GOT ONE
00F3'   E1              ..RET:  POP H
00F4'   DDE1                    POP X
00F6'   C1                      POP B
00F7'   C9                      RET
00F8'   C1              ..THERE: POP B          ;GET ORIG HL
00F9'   C5                      PUSH B          ;ANS SAVE IT
```

```
00FA'  E5                  PUSH H
00FB'  AF                  XRA A
00FC'  ED42                DSBC B          ;HL=BC?
00FE'  E1                  POP H
00FF'  20EA                JRNZ ..MORE     ;NO
0101'  CD 0227'            CALL NXTLNK
0104'  DD7503              MOV $NLINK(X),L
0107'  DD7404              MOV $NLINK+1(X),H       ;YUP
010A'  18E7                JMPR ..RET
                    ; *ISTERM--RETURNS IF NULL, ETC

                    ;ALLOWS NULL, NL, ',', ';', AND ' ' AS
                    ;VALID LINE TERMIES AND JUMPS OVER THEM
                    ;ALSO CALLS PUTLPT FOR YOU
                    ;ER.TERM IS OTHER TERMIES SEEN

010C'  7E           ISTERM: MOV A,M
010D'  B7                   ORA A
010E'  C8                   RZ
010F'  23                   INX H
0110'  CD 0243'             CALL PUTLPT
0113'  FE0A                 CPI NL
0115'  C8                   RZ
0116'  FE2C                 CPI ','
0118'  C8                   RZ
0119'  FE20                 CPI ' '
011B'  C8                   RZ
011C'  FE3B                 CPI ';'
011E'  C8                   RZ
011F'  2B                   DCX H
0120'  2B                   DCX H           ;SEE IF NULL BEFORE IT
0121'  7E                   MOV A,M         ;IN CASE OF SKIP -999
0122'  B7                   ORA A           ;WILL BE -> BEFORE STRING
0123'  23                   INX H           ;POINT PAST NULL
0124'  7E                   MOV A,M         ;SO A DOESN'T HAVE NULL
0125'  CD 0243'             CALL PUTLPT     ;SAVE PTR
0128'  C8                   RZ              ;OR AT ERROR
0129'  CD 0000'             CALL WARNFF     ;WARN THEM
012C'  544F4F204D55         .ASCIZ /TOO MUCH ON LINE: /
013F'  CD 0000:05           CALL ARGLIN     ;PRINT IT
0142'  C3 002C'             JMP NLPNT


                    ; *ISNL--RETURNS Z IF CR OR ;  *
                    ;KISNL DOES KLB FIRST

                    ;IN     HL->CHARACTER
                    ;OUT    Z SET IF CHARACTER IS CR OR ; OR NULL

0145'  CD 015B'     KISNL:  CALL KLB        ;DO KLB FIRST
0148'  7E           ISNL:   MOV A,M
0149'  B7                   ORA A
014A'  C8                   RZ
014B'  FE3B                 CPI ';'
014D'  C8                   RZ
```

```
014E'  FE0A                CPI NL
0150'  C9                  RET


                    ;  *SCANNL--SCANS LINE TILL NEXT NL*

                    ;IN     HL->CHARACTER
                    ;OUT    HL->PAST NL

0151'  7E           SCANNL: MOV A,M
0152'  A7                   ANA A           ;IS NULL?
0153'  C8                   RZ
0154'  23                   INX H
0155'  FE0A                 CPI NL          ;IS NL
0157'  C8                   RZ
0158'  18F7                 JMPR SCANNL     ;LOOK MORE

                    ;  *KLB--KILLS OFF BLANX*

                    ;IN     HL->CHARACTER, MAYBE BLANK
                    ;OUT    HL->FIRST NON-BLANK CHARACTER
                    ;KLBINC INCREMENTS HL FIRST

015A'  23           KLBINC: INX H
015B'  7E           KLB:    MOV A,M
015C'  FE20                 CPI BLANK
015E'  C0                   RNZ
015F'  18F9                 JMPR KLBINC

                    ; *CLEARIT--CLEARS FROM HL FOR BC BYTES*
0161'  E5           CLEARIT: PUSH H
0162'  D5                   PUSH D
0163'  54                   MOV D,H
0164'  5D                   MOV E,L
0165'  13                   INX D
0166'  3600                 MVI M,0
0168'  EDB0                 LDIR
016A'  D1                   POP D
016B'  E1                   POP H
016C'  C9                   RET


                    ;  *SCANLAB--SKIPS OVER LABELS*

016D'  CD 025F'     SCANLAB:        CALL CARTYP     ;SET CC'S
0170'  C8                   RZ              ;RET IF DELIM
0171'  23                   INX H
0172'  18F9                 JMPR SCANLAB

                    ; *ACOPY--COPIES HL TO DE UNTIL DELIM IN A SEEN*

0174'  BE           ACOPY:  CMP M
0175'  2804                 JRZ ..OUT       ;COPY DELIM TOO
0177'  EDA0                 LDI
0179'  18F9                 JMPR ACOPY
```

```
017B'   EDA0            ..OUT:  LDI
017D'   C9                      RET

                        ; *SCANB--SCANS PAST NEXT BLANK OR TO NL*

                        ;SCANB
                        ;IN--HL->CHARACTERS WITH POSSIBLE LEADING BLANX
                        ;OUT--HL->AT BLANK FOLLOWING NONBLANK CHARS
                        ;       (I.E., IT SKIPS OVER A BUNCH OF NON-BLANKS)
                        ;Z IS SET IF POINTING AT A NL, ; OR NULL
                        ;Z CLEAR IF -> AT A BLANK UPON EXIT

017E'   CD 0145'        SCANB:  CALL KISNL
0181'   C8                      RZ
0182'   23                      INX H
0183'   7E                      MOV A,M
0184'   FE20                    CPI BLANK
0186'   20F6                    JRNZ SCANB
0188'   B7                      ORA A
0189'   C9                      RET

                        ; *SWITCH--POPS ONE LEVEL IN A MACRO*
                        ;IN:    NUTTIN (USES MACTOP)
                        ;OUT:   HL->CODE
                        ;       A HAS RETURNED TYPE, DE HAS VALUE
                        ;        FROM $RVSTUFF
                        ;       OLD MIB IS ZAPPED AND CALLER RESTORED

018A'   C5              SWITCH: PUSH B          ;SAVE B
018B'   FDE5                    PUSH Y          ;SAVE Y TOO
018D'   2A 625E                 LHLD MACTOP     ;GET MIB
0190'   7C                      MOV A,H         ;IS ZERO?
0191'   B5                      ORA L           ;IF SO, IS FROM *-LEVEL
                                ZERROR ER.RET   ;AND IS BOOBOO[
0192'   2004            +       JRNZ ..0001
0194'   CD 0000:07      +       CALL ERRPGM
0197'   24              +       .BYTE ER.RET
0198'                   +..0001:]
0198'   FD2A 625E                LIYD MACTOP
019C'   FD7E1A                  MOV A,$GOSUB(Y) ;CHECK GOSUB COUNT
019F'   B7                      ORA A           ;IF #0, IS GOSUB RET
01A0'   202F                    JRNZ ..DEC      ;DO IT
01A2'   01 0013                 LXI B,$RVSTUFF  ;POINT IY TO GOODIES
01A5'   FD09                    DADY B
01A7'   CD 0000:09              CALL GETOPND    ;GET STUFF IN A&DE
01AA'   F5                      PUSH PSW
01AB'   D5                      PUSH D          ;SAVE A,DE FOR EXIT
01AC'   7E                      MOV A,M         ;IS $KEYBLK?
01AD'   FE22                    CPI $MIBBLK     ;INSTEAD OF $MIBBLK
01AF'   2003                    JRNZ ..DODD     ;IF SO, DON'T DELETE MIB
01B1'   CD 0000:06              CALL  DELMIB
01B4'   01 000D         ..DODD: LXI B,$CALLER   ;POINT TO CALLER
01B7'   CD 022A'                CALL NXTABC     ;...
01BA'   22 625E                 SHLD MACTOP     ;STORE IT
01BD'   01 0005         ..DONT: LXI B,$LINPTR   ;POINT TO CALLER RETURN
```

```
01C0'   CD 022A'          CALL NXTABC       ;LINE
01C3'   D1          ..GDON: POP D             ;GET RETURNED VALUES
01C4'   F1                  POP PSW           ;BACK--WAS A=0?
01C5'   B7                  ORA A
01C6'   2005                JRNZ ..BYE        ;SOMETHING WAS PASSED!
01C8'   3E04        ..GOSX: MVI A,$IVAL       ;DEFAULT=SUCCEED
01CA'   11 0001             LXI D,1
01CD'   FDE1        ..BYE:  POP Y
01CF'   C1                  POP B
01D0'   C9                  RET               ;GO BACK
01D1'   FD351A      ..DEC:  DCR $GOSUB(Y)     ;DECREMENT COUNT
01D4'   EB                  XCHG              ;DE HAD RETURN POINTER
01D5'   18F1                JMPR ..GOSX       ;ALL DONE

                    ;  *STREQK--CHECKS IF TWO STRINGS EQUAL TO DELIM*

                    ;IN     HL->FIRST CHAR OF STR1
                    ;       DE->FIRST CHAR OF STR2
                    ;
                    ;OUT    DE,HL SAME
                    ;       Z SET IF STRINGS = TO DELIM
                    ;       Z CLEAR IF STRINGS NOT =

01D7'   D5          STREQK: PUSH D            ;SAVE DE
01D8'   E5                  PUSH H            ;AND HL
01D9'   1A          ..STRO: LDAX D            ;GET CHAR DE POINTS AT
01DA'   BE                  CMP M             ;IS SAME AS HL POINTEE?
01DB'   2009                JRNZ ..STR1       ;IF NE, CHECK CARTYPS
01DD'   CD 025F'            CALL CARTYP       ;CHECK TYPE
01E0'   2809                JRZ ..STRQ        ;IF DELIM
01E2'   23                  INX H
01E3'   13                  INX D
01E4'   18F3                JMPR ..STRO       ;TRY AGAIN
01E6'   CD 025F'    ..STR1: CALL CARTYP       ;DELIM?
01E9'   2004                JRNZ ..STR2       ;...
01EB'   EB          ..STRQ: XCHG              ;TRY OTHER ONE FOR DELIM
01EC'   CD 025F'            CALL CARTYP       ;SETS CC'S OK
01EF'   E1          ..STR2: POP H
01F0'   D1                  POP D
01F1'   C9                  RET               ;SIN(ARA)


                    ;  *FINDLAB--SEARCHES FOR A LABEL ON A LINE*

                    ;IN     HL->LABEL SOMEWHERE
                    ;OUT    HL->LINE BEGINNING WITH LABEL
                    ;               OR ER.LAB GENERATED

01F2'   C5          FINDLAB:        PUSH B
01F3'   D5                  PUSH D
01F4'   FDE5                PUSH Y
01F6'   FD2A 625E           LIYD MACTOP
01FA'   EB                  XCHG
01FB'   FD6604              MOV H,$STRPTR+1(Y)
01FE'   FD6E03              MOV L,$STRPTR(Y)
```

```
0201'   01 000A              LXI B,$SASCII   ;POINT AT STRING
0204'   09                   DAD B
0205'   060A                 MVI B,NL        ;LOAD B WITH NL
0207'   CD 01D7'    ..FINO: CALL STREQK
020A'   280E                 JRZ ..FINX      ;OKY--GOT IT
020C'   7E           ..SCAN: MOV A,M         ;IS NULL?
020D'   23                   INX H           ;ADVANCE HL
020E'   B7                   ORA A
                             ZERROR ER.LABE
020F'   2004         +       JRNZ ..0002
0211'   CD 0000:07   +       CALL ERRPGM
0214'   25           +       .BYTE ER.LAB
0215'                +..0002:]
0215'   B8                   CMP B           ;IS A NL?
0216'   20F4                 JRNZ ..SCAN     ;NOPE, LOOK MORE
0218'   18ED                 JMPR ..FINO     ;CHECK FOR LABEL
021A'   CD 016D'    ..FINX: CALL SCANLAB    ;POINT PAST LABEL
021D'   FDE1                 POP Y
021F'   D1                   POP D
0220'   C1                   POP B
0221'   C9                   RET


                     ;  *NXTPTR, NXTLNK, NXTABC--DOES MOV HL,(HL)*
                     ;       NOTE:   BC IS CLOBBERED!

                     ;       NXTPTR DOES MOV HL,(HL)
                     ;       NXTLNK DOES MOV HL,$NLINK(HL)
                     ;       NXTVAL DOES MOV HL,$NVALUE(HL)
                     ;       NXTABC DOES MOV HL,BC(HL)

                     ;IN     HL->SOMETHING
                     ;OUT    HL CONTAINS THAT STUFF
                     ;       AND Z IS SET IF HL=0

0222'   01 0005      NXTVAL: LXI B,$NVALUE
0225'   1803                 JMPR NXTABC
0227'   01 0003      NXTLNK: LXI B,$NLINK    ;NAME LINK
022A'   09           NXTABC: DAD B   ;ADD IN BC TO HL
022B'   D5           NXTPTR: PUSH D
022C'   5E                   MOV E,M         ;GET IT
022D'   23                   INX H           ;SLOW STUFF
022E'   56                   MOV D,M         ;GET SECOND HALF
022F'   7A                   MOV A,D         ;CHECK FER ZERO
0230'   B3                   ORA E           ;?
0231'   EB                   XCHG            ;SETS CC'S
0232'   D1                   POP D           ;EXIT
0233'   C9                   RET

                     ;  *GETLPT--GETS LINPTR INTO HL FROM MACTOP*

0234'   FDE5         GETLPT: PUSH Y
0236'   FD2A 625E             LIYD MACTOP     ;GET MACTOP ADDR
023A'   FD6E05               MOV L,$LINPTR(Y)
```

```
023D'   FD6606              MOV H,$LINPTR+1(Y)
0240'   FDE1        GETLPX: POP Y
0242'   C9                  RET

                    ; *PUTLPT--PUTS LINPTR FROM HL INTO MACTOP*

0243'   FDE5        PUTLPT: PUSH Y
0245'   FD2A 625E           LIYD MACTOP
0249'   FD7505              MOV $LINPTR(Y),L
024C'   FD7406              MOV $LINPTR+1(Y),H
024F'   18EF                JMPR GETLPX


                    ; *INC5IY--ADDS 5 TO IY*

0251'   C5          INC5IY: PUSH B
0252'   01 0005             LXI B,5
0255'   FD09        IYOUT:  DADY B
0257'   C1                  POP B
0258'   C9                  RET

                    ; *DEC5IY--SUBS 5 FROM IY*

0259'   C5          DEC5IY: PUSH B
025A'   01 FFFB             LXI B,-5
025D'   18F6                JMPR IYOUT

                    ; *CARTYP--SETS CC'S ACCORDING TO WHAT HL POINTS AT*

                    ;IN     HL->CHARACTER
                    ;OUT    HL->CHARACTER
                    ;       Z CLEAR, C CLEAR IF ALPHA
                    ;       Z CLEAR, C SET   IF NUMERIC
                    ;       Z SET    C CLEAR IF DELIM
                    ;CARTYA:        USES A INSTEAD OF M


025F'   7E          CARTYP: MOV A,M         ;GET CHARACTER
0260'   FE24        CARTYA: CPI '$'         ;$ IS ALPHA
0262'   2822               JRZ ..CA        ;FOR US
0264'   FE30               CPI '0'
0266'   FA 0282'           JM ..CD         ;IS DELIM
0269'   FE3A               CPI '9'+1
026B'   FA 0284'           JM ..CN         ;IS NUMERIC
026E'   FE41               CPI 'A'
0270'   FA 0282'           JM ..CD         ;IS DELIM
0273'   FE5B               CPI 'Z'+1
0275'   FA 0286'           JM ..CA         ;IS ALPHA
0278'   FE61               CPI 141Q        ;LITTLE A
027A'   FA 0282'           JM ..CD         ;IS DELIM
027D'   FE7C               CPI 141Q+27     ;LITLE Z+1
027F'   FA 0286'           JM ..CA         ;IS ALPHA
0282'   AF          ..CD:   XRA A           ;SET Z,CLEAR CARRY
0283'   C9                  RET
0284'   37          ..CN:   STC             ;SET CARRY,Z IS CLEAR
```

```
0285'   C9                      RET
0286'   A7              ..CA:   ANA A           ;CLEAR CARRY,
                                                ;LEAVE Z CLEAR
0287'   C9                      RET

0288'                   SUBEND:
                                .END
```

+++++ SYMBOL TABLE +++++

AASN	005F		ABS	008C′		ACOPY	0174′	I	ALLOC	0000:04	X
ARGLIN	0000:05	X	ARGSTK	60CC		BACKGR	65CC		BLANK	0020	
BOTRAM	6000		BOTTOM	64A9		CARTYA	0260′	I	CARTYP	025F′	I
CHARSL	65DD		CLEARI	0161′	I	CLEARS	60CA		CNTRL	000C	
CNTRLC	65CD		CNTRLO	65D9		CNTRLU	0015		CNTRLZ	65E4	
CPLARE	611C		CPLSIZ	0140		CR	000D		CSBLOK	668D	
CSFLAG	6260		CURCX	6814		CURCY	6816		CURREN	64A5	
C.CO	0011		C.C1	0012		C.CX	000B		C.CY	000D	
C.DP	0013		C.ST	0002		C.X	0003		C.XF	000F	
C.XS	0007		C.Y	0005		C.YF	0010		C.YS	0009	
DDTON	65CE		DEC5IY	0259′	I	DELMIB	0000:06	X	DEVBL	6589	
DEVCL0	6579		DEVCL1	657B		DEVCL2	657D		DEVCL3	657F	
DEVCL4	6581		DEVCL5	6583		DEVCL6	6585		DEVCL7	6587	
DEVFB	65CB		DEVHCB	658F		DEVMO	658B		DEVNM	65B7	
DEVNT	658D		DEVTNA	65BB		DEVTNB	65BF		DEVTNC	65C3	
DEVVA	65BD		DEVVAR	6579		DEVVB	65C1		DEVVBL	6591	
DEVVC	65C5		DEVVD	65C9		DEVVN	65B9		DEVVS	65C7	
DEVXCD	65B3		DEVYCD	65B5		DOLPLH	62E8		DOLPPT	62EA	
DUMBST	6577		EDBCNT	64AD		EDCCNT	64A7		EDLONG	6812	
EDMODE	681C		EDNAME	64A1		EDNCX	680C		EDNCY	680E	
EDNEWS	64A5		EDOCX	6808		EDOCY	680A		EDPN	6806	
EDPO	6804		EDPTRC	64AB		EDPTRL	64A9		EDSTR	681A	
ERABIT	0002		ERRPGM	0000:07	X	ER.ARA	002F		ER.ARG	0034	
ER.ASN	0015		ER.BOX	001A		ER.CHN	0002		ER.CMD	001F	
ER.CNV	0016		ER.COR	001B		ER.CTL	0036		ER.DEL	0026	
ER.DIM	0030		ER.DIV	0018		ER.DP	0037		ER.DSK	0019	
ER.EDT	0035		ER.FMT	0038		ER.FNF	001C		ER.FOR	0028	
ER.IMP	0003		ER.LAB	0025		ER.MAC	0022		ER.NAE	002D	
ER.NAM	0029		ER.NEG	003A		ER.NOT	0023		ER.NUL	0039	
ER.NUM	002B		ER.NXT	001D		ER.OFL	0017		ER.OPN	0014	
ER.OVE	001E		ER.PAR	002A		ER.REN	002C		ER.RET	0024	
ER.SEP	0021		ER.SNP	0031		ER.SPC	002E		ER.STK	0004	
ER.SW	0032		ER.TER	0020		ER.UFL	0033		ER.UNF	0027	
EXTDEL	002E		E.HVAL	0002		E.LVAL	0001		E.SIZ	0005	
E.TYP	0000		E.VAL	0001		FCNTH	65D2		FCNTI	65D3	
FCNTJ	65D4		FCNTK	65D5		FCNTL	65D6		FCNTV	65E0	
FCNTY	65E3		FINDLA	01F2′	I	FIRST	64A3		FLAGS	65CB	
FOREGR	65D0		FRAGSI	0400		FREE	0000:08	X	FREELS	65E5	
FSTDOL	648C		FSTINT	62EC		FWDPTR	65E7		GETLPT	0234′	I
GETLPX	0240′		GETOPN	0000:09	X	HCAREA	6593		HEXBYT	00B8′	I
HEXHL	00B7′	I	INC5IY	0251′	I	INCRO	65E9		INHEX	0096′	I
ISNL	0148′	I	ISTERM	010C′	I	IYOUT	0255′		JUNK	6542	
KBLOCK	65F1		KEYFLG	67FF		KEYPTK	6533		KEYTRK	67F7	
KISNL	0145′	I	KLB	015B′	I	KLBINC	015A′		LF	000A	
LISTON	65E2		MACSTU	6536		MACTOP	625E		MAXFRG	0040	
MNMX	65EB		NBLKB	0000		NBLKM	0001		NEWBOT	6810	
NL	000A		NLPNT	002C′	I	NSADDR	6802		NUMBUF	64A3	
NXTABC	022A′	I	NXTLNK	0227′	I	NXTPTR	022B′	I	NXTVAL	0222′	I
OLDCHR	64A0		OLDCUR	649F		OLDKEY	649D		OLDXY	65EF	
ONEBUF	6729		OPRL	0014		OPRSP	655B		OPRSTK	6547	
OPRSZ	655D		OUTCH	0000:0A	X	OUTOFF	62E7		PCNT	655E	
PDECA	0036′	I	PDECB1	003D′		PDECBC	003C′	I	PIXVAL	65ED	
POINTE	64A7		PONOFF	65DA		PRINTA	001B′	I	PRINTF	000D′	I
PRINTH	001A′	I	PRINTR	6540		PUTLPT	0243′	I	RAMEND	7FFF	
RAMSTR	6900		RANSHT	655F		RMDTMP	6538		RUBACK	0005	

SUB - ZGRASS SUBROUTINES
+++++ SYMBOL TABLE +++++

RUBOUT	007F		SAVESP	625C		SCANB	017E′	I	SCANLA	016D′	I
SCANNL	0151′	I	SCNEED	0004		SCRWIN	67CD		SOPRSP	653D	
SOPRSZ	653F		STACK	60C8		STAKTO	6000		STREQK	01D7′	I
STRSIZ	6800		SUBEND	0288′		SUBSTU	6534		SWITCH	018A′	I
TAB	0009		TABPN1	0032′		TABPNT	0030′	I	TAPBUF	64B3	
TAPCON	64AF		TAPPRO	64B1		TBFEND	6533		TEMPCH	00CF′	I
TEMPHD	60CA		TEMPS	62E5		TEMPUN	00E4′	I	TMPARG	67AD	
TOP	6818		TTYBEG	6261		TTYEND	62E1		TTYINT	62E3	
TTYPTR	62E1		TXTWIN	67E2		UARTFL	649C		USREND	65F1	
V3PTR	6573		VDCHAR	649E		VDNLF	65CF		VIPLH	6545	
VOICE0	6563		WARNFF	0000′	I	WRMODE	65EE		ZGIM2	0001	
ZGREND	681D		$AADR	0010		$ADDRF	0007		$ADDRI	0005	
$ADDRS	0009		$ANY	001A		$ANYNA	FFFC		$ANYVA	FFFE	
$ARGPT	0011		$BGPTR	000F		$BNDL	0007		$CALLE	000D	
$CMDAD	0018		$CPLBL	002A		$CSBLO	0028		$DATAP	0007	
$DOLDE	0001		$DOLOO	0002		$DVAL	0000		$END	001C	
$FADR	000E		$FLAGS	0002		$FORBL	0024		$FORPT	000B	
$FVAL	0006		$GOSUB	001A		$IADR	000C		$INPBU	0018	
$INPPT	0016		$IVAL	0004		$KEYBL	0026		$LENGT	0001	
$LINPT	0005		$LOCPT	0009		$MIBBL	0022		$MIBEN	001B	
$NAMAD	000A		$NAME	000A		$NASCI	0009		$NDEL	0080	
$NLINK	0003		$NULL	0002		$NVALU	0005		$REPEA	001E	
$RVSTU	0013		$SAME	0020		$SASCI	000A		$SLEN	0006	
$STRAD	0008		$STRPT	0003		$TAF	0000		$TYPE	0000	
$USE	0005		.BLNK.	0000:03	X	.DATA.	0000*	X	.PROG.	0288′	X

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                       ***
@                     *******
@                          *
@                        *
@                       *
@                      *
@                     *
@                     *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                 THE VERY BEST"
@
@              ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                    (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
          .LINK
          .PREL
          .IDENT  DOL
          .INTERN DELMIB
          .INTERN DOCHK
          .INTERN KILMIB
          .INTERN MAKMIB
          .INTERN STOP
 ;
          .EXTERN ABORT
          .EXTERN ALLOC
          .EXTERN ALLOCD
          .EXTERN ARG
          .EXTERN CZCHEK
          .EXTERN CLEARIT
          .EXTERN DEC5IY
          .EXTERN DISPLAY
          .EXTERN ERRPGM
          .EXTERN EVAL
          .EXTERN EVALARG
          .EXTERN FREE
          .EXTERN FREEALL
          .EXTERN GETLPT
          .EXTERN GETOPND
          .EXTERN INC5IY
          .EXTERN INCUSE
          .EXTERN ISTERM
          .EXTERN KBCNLP
          .EXTERN KISNL
          .EXTERN LINEP
          .EXTERN NXTABC
          .EXTERN NXTLNK
          .EXTERN NXTPTR
```

```
                                .EXTERN NXTVAL
                                .EXTERN PUTOPND
                                .EXTERN PUTLPT
                                .EXTERN SDEL
                                .EXTERN SNAP


                        ; DOLOOPED MACRO PROCESSING LOOP
                        ; CALLED WHILE WAITING FOR HUMAN INPUT
                        ; THIS ROUTINE WILL EXECUTE ONE STATEMENT FROM
                        ; EACH DOLOOPED MACRO ON THE DOLOOP LIST
                        ; AND THEN RETURN
0000'  2A 625E          DOCHK:  LHLD    MACTOP
0003'  E5                       PUSH    H
0004'  2A 62E8                  LHLD    DOLPLH  ; START LIST AT TOP
0007'  DD21 62D9                LXI     X,DOLPLH-$BGPTR
000B'  22 62EA          ..NEXT: SHLD    DOLPPT  ; STUFF INTO WORKING PTR
000E'  7C                       MOV     A,H
000F'  B5                       ORA     L       ;IS PTR NULL?

0010'  2846                     JRZ     ..DONE  ; KICKOUT IF DONE
                        ; SHOULD WE DELETE THIS MIB?
0012'  23                       INX     H       ; MOVE TO FLAG BYTE
0013'  23                       INX     H
0014'  CB4E                     BIT     $DOLDEL,M
0016'  2B                       DCX     H
0017'  2B                       DCX     H
0018'  2813                     JRZ     ..NDIE          ; NO
                        ; DELETE BIT IS SET - SO KILL THIS GUY OFF
001A'  E5                       PUSH    H
001B'  01 000F                  LXI     B,$BGPTR        ;POINT AT NEXT
001E'  CD 0000:19               CALL NXTABC
0021'  DD7410                   MOV     $BGPTR+1(X),H   ; SET PREVIOUS
0024'  DD750F                   MOV     $BGPTR(X),L     ;TO POINT AT US
0027'  E1                       POP     H               ; RELEASE MEMORY
0028'  CD 0138'                 CALL    DELMIB
002B'  1823                     JMPR    ..NEXM          ;  NEXT MACRO
                        ; GOOD DOLOOPED MACRO
002D'  22 625E          ..NDIE: SHLD    MACTOP  ; SET THIS GUY AS TOP
0030'  CD 0000:11               CALL    GETLPT  ; GET STRING SCANNING PTR
0033'  CD 0000:18       ..LC:   CALL    LINEP   ; EVAL A LINE
0036'  CD 0000:08               CALL CZCHEK     ;DEBUG/CNTRLZ?
0039'  DD2A 62EA                LIXD    DOLPPT  ;IX->CURRENT MIB

                        ; DID THIS GUY RETURN?
003D'  CD 0000:15               CALL ISTERM     ;PROPER TERMY?
0040'  B7                       ORA A           ;NULL?
0041'  200D                     JRNZ ..NEXM     ;NOP-NEXT MACRO
                        ; YES HE DID - RESET STRING PTR TO FIRST CHAR
0043'  DD6604                   MOV     H,$STRPTR+1(X)
0046'  DD6E03                   MOV     L,$STRPTR(X)
0049'  01 000A                  LXI     B,$SASCII
004C'  09                       DAD     B
                        ; STUFF POINTER BACK INTO MIB
004D'  CD 0000:1E               CALL PUTLPT
```

```
                        ; MOVE TO NEXT MIB ON CHAIN AND TRY TO DOIT
0050'  DD6610           ..NEXM: MOV     H,$BGPTR+1(X)
0053'  DD6E0F                   MOV     L,$BGPTR(X)
0056'  18B3                     JMPR    ..NEXT
0058'  E1               ..DONE: POP     H
0059'  22 625E                  SHLD    MACTOP
005C'  C9                       RET



                        ; COMMAND TO UNDOLOOP A MACRO
                        ; THIS GUY JUST SETS A DIE! FLAG
                        ; WHICH THE DOLOOP DISPATCHING LOOP LOOKS
                        ; AT TO DO THE ACTUAL DELETION
                        M.CMD[STOP,..UNLX,0,SNAP,..UNLA,..UNLS]I
005D'  18               +STOP:  .BYTE $CMDADR
005E'  05               +       .BYTE (..UNLX-STOP)/16+1
005F'  00               +       .BYTE 0
0060'  0000:20          +       .WORD SNAP
0062'  006C'            +       .WORD ..UNLS
0064'  0075'            +       .WORD ..UNLA
0066'  53544F5000       +       .ASCIZ /STOP/
                        +]
006B'  00                       .BYTE 0         ;NO SWITCHES
006C'  CD 0000:17       ..UNLS: CALL KISNL      ;IS STOP ALONE?
006F'  CA 0000:04               JZ ABORT
0072'  CD 0000:07               CALL    ARG
0075'  081E             ..UNLA: .BYTE   $STRADR,$REPEAT
0077'  E5                       PUSH    H
0078'  FD7E00           ..UNLL: MOV     A,0(Y)          ; MORE TO DO
007B'  A7                       ANA     A
007C'  282C                     JRZ ..DOUT              ;DONE
007E'  FD5602                   MOV     D,2(Y)          ; DE=STRING ADDR
0081'  FD5E01                   MOV     E,1(Y)
0084'  2A 62E8                  LHLD    DOLPLH          ;  LIST TOP
0087'  7C               ..ZAP:  MOV     A,H
0088'  B5                       ORA     L
0089'  2810                     JRZ     ..NULL          ; NULL IF NOFIND
008B'  23                       INX     H               ; ELSE STEP
                                                        ;TO STR ADDR
                        ; TO STRING ADDRESS
008C'  23                       INX     H
008D'  23                       INX     H
008E'  7E                       MOV     A,M             ; COMPARE TO
008F'  23                       INX     H               ;SEARCH TARGET
0090'  BB                       CMP     E
0091'  200D                     JRNZ    ..NONO          ;NO MATCH
0093'  7E                       MOV     A,M
0094'  BA                       CMP     D
0095'  2009                     JRNZ    ..NONO
                        ; WE GOT A LIVE ONE
0097'  2B                       DCX     H               ; BACK TO FLAGS
0098'  2B                       DCX     H
0099'  CBCE                     SET     $DOLDEL,M       ; SAY DELETE HIM
```

```
009B'  CD 0000:13    ..NULL: CALL INC5IY
009E'  18D8                  JMPR    ..UNLL
                     ; MOVE ON DOWN THE LIST...
00A0'  01 000B       ..NONO: LXI     B,$BGPTR-4      ; NEXT PTR
00A3'  09                    DAD     B
00A4'  7E                    MOV     A,M
00A5'  23                    INX     H
00A6'  66                    MOV     H,M
00A7'  6F                    MOV     L,A
00A8'  18DD                  JMPR    ..ZAP
00AA'  E1            ..DOUT: POP H
00AB'  C9                    RET
00AC'                ..UNLX:
                     ; PROCEDURE TO BUILD AN MIB
                     ; NEEDS:
                     ;HL->DELIM FOLLOWING MACRO NAME IN LINE
                     ; RETURNS:
                     ; IX - MIB ADDRESS
                     ; HL->LINE TERMINATOR
                     ; MIB SETUP WITH ARGS PARSED AND STUCK AT
                     ; $MIBEND

00AC'  C5            MAKMIB: PUSH B
00AD'  D5                    PUSH D
00AE'  E5                    PUSH H
00AF'  D5                    PUSH D          ;SAVE NAMEBLOCK FWA
00B0'  3E05                  MVI A,5         ;GET 6 BLOX
00B2'  CD 0000:05            CALL ALLOC      ;GET MIB SPACE
00B5'  E5                    PUSH H          ;SAVE FWA
00B6'  3622                  MVI M,$MIBBLK   ;TAG IT
00B8'  23                    INX H           ;SKIP TAG
00B9'  23                    INX H           ;AND LENGTH
00BA'  01 001B               LXI B,$MIBEND   ;CLEAR STUFF
00BD'  CD 0000:09            CALL CLEARIT
00C0'  DDE1                  POP X           ;GET MIBFWA BACK
00C2'  E1                    POP H           ;GET NAMEBLOCK FWA
00C3'  CD 0000:1C            CALL NXTVAL     ;POINT AT STRING
00C6'  54                    MOV D,H
00C7'  5D                    MOV E,L         ;INCREMENT USE COUNT
00C8'  CD 0000:14            CALL INCUSE
00CB'  DD7503                MOV $STRPTR(X),L        ;POINT STRPTR
00CE'  DD7404                MOV $STRPTR+1(X),H
00D1'  01 000A               LXI B,$SASCII   ;POINT ASCII
00D4'  09                    DAD B
00D5'  DD7505                MOV $LINPTR(X),L
00D8'  DD7406                MOV $LINPTR+1(X),H
00DB'  E1                    POP H           ;GET CALLING LINE PTR BACK
00DC'  CD 0000:13            CALL    INC5IY  ; ** FIX **
00DF'  FDE5                  PUSH Y          ;SAVE IY STACK
00E1'  DDE5                  PUSH X          ;MIBFWA
00E3'  FDE1                  POP Y           ;IN Y
00E5'  01 001B               LXI B,$MIBEND
00E8'  FD09                  DADY B          ;WHERE TO STICK IT
00EA'  FDE5                  PUSH Y          ;PUT ARGPTR IN
00EC'  C1                    POP B           ;MOV $(X),Y
```

```
00ED'   DD7111              MOV $ARGPTR(X),C ;MOV $+1(X),I
00F0'   DD7012              MOV $ARGPTR+1(X),B
00F3'   FD360000    ..MORE: MVI 0(Y),$TAF    ;TERMINATE LIST SO FAR
00F7'   CD 0000:16          CALL KBCNLP      ;ARGS??
00FA'   2816                JRZ ..DONE       ;NONE (LEFT)
00FC'   FDE3                XTIY             ;SETUP FOR EVAL
00FE'   CD 0000:0E          CALL EVALARG
0101'   CD 0000:12          CALL GETOPND     ;GET IT
0104'   FDE3                XTIY             ;MIBEND STACK BACK
0106'   CD 0000:1D          CALL PUTOPND     ;STICK IT THERE
0109'   CD 0000:13          CALL INC5IY      ;POINT TO NEXT
010C'   FD360000            MVI 0(Y),$TAF    ;TERMINATE LIST
0110'   18E1                JMPR ..MORE      ;LOOK FER MORE
0112'   FDE5        ..DONE: PUSH Y           ;SETUP FOR ALLOCD
0114'   D1                  POP D
0115'   13                  INX D            ;POINT 1 PAST
0116'   E5                  PUSH H           ;SAVE CURRENT LINE PTR
0117'   DDE5                PUSH X           ;GET MIBFWA
0119'   E1                  POP H            ;INTO HL
011A'   CD 0000:06          CALL ALLOCD      ;RELEASE EXTRA
011D'   E1                  POP H            ;RESTORE H
011E'   FDE1                POP Y
0120'   CD 0000:0A          CALL    DEC5IY
0123'   D1                  POP D
0124'   C1                  POP B
0125'   C9                  RET              ;LEAVE WITH IX->MACTOP

                    ;KILMIB--CHAINS MIB'S BACKWARDS, DELETING ALL

                    ;HL IS USUALLY CONSIDERED CLOBBERED
                    ;AND SO IS BC

0126'   7C         KILMIB: MOV A,H          ;HL->FIRST MIB TO GO
0127'   B5                 ORA L
0128'   C8                 RZ
0129'   7E                 MOV A,M          ;CHECK IF $MIBBLK
012A'   FE22               CPI $MIBBLK      ;IF NOT, MAYBE ALREADY
012C'   C0                 RNZ              ;GONE!
012D'   CD 0138'           CALL DELMIB      ;DELETE IT AND HANGIES
0130'   01 000D            LXI B,$CALLER
0133'   CD 0000:19         CALL NXTABC      ;CHAIN TO PREV CALLER
0136'   18EE               JMPR KILMIB      ;TRY TRY AGAIN


                    ;DELMIB--DELETES MIB AND ALL STUFF HANGING OFF IT

0138'   C5         DELMIB: PUSH B
0139'   E5                 PUSH H
013A'   01 000B            LXI B,$FORPTR    ;DELETE
013D'   CD 0000:19         CALL NXTABC
0140'   CD 0000:10         CALL FREEALL     ;ALL FORBLOX
0143'   E1                 POP H
0144'   E5                 PUSH H
0145'   01 0003            LXI B,$STRPTR    ;DECUSE COUNT
0148'   CD 0000:19         CALL NXTABC      ;POINT AT STRING
```

```
014B'   CD 0000:1F          CALL SDEL       ;DEC IT
014E'   E1                  POP H           ;RESTORE H
014F'   C1                  POP B
0150'   C3 0000:0F          JMP FREE        ;THEN THE MIB ITSELF

                    .END
```

DOL - DOL--DOLOOP STUFF AND MIB STUFF
+++++ SYMBOL TABLE +++++

```
AASN    005F          ABORT   0000:04 X     ALLOC   0000:05 X     ALLOCD  0000:06 X
ARG     0000:07 X     ARGSTK  60CC          BACKGR  65CC          BLANK   0020
BOTRAM  6000          BOTTOM  64A9          CHARSL  65DD          CLEARI  0000:09 X
CLEARS  60CA          CNTRL   000C          CNTRLC  65CD          CNTRLO  65D9
CNTRLU  0015          CNTRLZ  65E4          CPLARE  611C          CPLSIZ  0140
CR      000D          CSBLOK  668D          CSFLAG  6260          CURCX   6814
CURCY   6816          CURREN  64A5          CZCHEK  0000:08 X     C.C0    0011
C.C1    0012          C.CX    000B          C.CY    000D          C.DP    0013
C.ST    0002          C.X     0003          C.XF    000F          C.XS    0007
C.Y     0005          C.YF    0010          C.YS    0009          DDTON   65CE
DEC5IY  0000:0A X     DELMIB  0138'    I    DEVBL   6589          DEVCL0  6579
DEVCL1  657B          DEVCL2  657D          DEVCL3  657F          DEVCL4  6581
DEVCL5  6583          DEVCL6  6585          DEVCL7  6587          DEVFB   65CB
DEVHCB  658F          DEVMO   658B          DEVNM   65B7          DEVNT   658D
DEVTNA  65BB          DEVTNB  65BF          DEVTNC  65C3          DEVVA   65BD
DEVVAR  6579          DEVVB   65C1          DEVVBL  6591          DEVVC   65C5
DEVVD   65C9          DEVVN   65B9          DEVVS   65C7          DEVXCD  65B3
DEVYCD  65B5          DISPLA  0000:0B X     DOCHK   0000'    I    DOLPLH  62E8
DOLPPT  62EA          DUMBST  6577          EDBCNT  64AD          EDCCNT  64A7
EDLONG  6812          EDMODE  681C          EDNAME  64A1          EDNCX   680C
EDNCY   680E          EDNEWS  64A5          EDOCX   6808          EDOCY   680A
EDPN    6806          EDPO    6804          EDPTRC  64AB          EDPTRL  64A9
EDSTR   681A          ERABIT  0002          ERRPGM  0000:0C X     ER.ARA  002F
ER.ARG  0034          ER.ASN  0015          ER.BOX  001A          ER.CHN  0002
ER.CMD  001F          ER.CNV  0016          ER.COR  001B          ER.CTL  0036
ER.DEL  0026          ER.DIM  0030          ER.DIV  0018          ER.DP   0037
ER.DSK  0019          ER.EDT  0035          ER.FMT  0038          ER.FNF  001C
ER.FOR  0028          ER.IMP  0003          ER.LAB  0025          ER.MAC  0022
ER.NAE  002D          ER.NAM  0029          ER.NEG  003A          ER.NOT  0023
ER.NUL  0039          ER.NUM  002B          ER.NXT  001D          ER.OFL  0017
ER.OPN  0014          ER.OVE  001E          ER.PAR  002A          ER.REN  002C
ER.RET  0024          ER.SEP  0021          ER.SNP  0031          ER.SPC  002E
ER.ST   0004          ER.SW   0032          ER.TER  0020          ER.UFL  0033
ER.UNF  0027          EVAL    0000:0D X     EVALAR  0000:0E X     EXTDEL  002E
E.HVAL  0002          E.LVAL  0001          E.SIZ   0005          E.TYP   0000
E.VAL   0001          FCNTH   65D2          FCNTI   65D3          FCNTJ   65D4
FCNTK   65D5          FCNTL   65D6          FCNTV   65E0          FCNTY   65E3
FIRST   64A3          FLAGS   65CB          FOREGR  65D0          FRAGSI  0400
FREE    0000:0F X     FREEAL  0000:10 X     FREELS  65E5          FSTDOL  648C
FSTINT  62EC          FWDPTR  65E7          GETLPT  0000:11 X     GETOPN  0000:12 X
HCAREA  6593          INC5IY  0000:13 X     INCRO   65E9          INCUSE  0000:14 X
ISTERM  0000:15 X     JUNK    6542          KBCNLP  0000:16 X     KBLOCK  65F1
KEYFLG  67FF          KEYPTK  6533          KEYTRK  67F7          KILMIB  0126'    I
KISNL   0000:17 X     LF      000A          LINEP   0000:18 X     LISTON  65E2
MACSTU  6536          MACTOP  625E          MAKMIB  00AC'    I    MAXFRG  0040
MNMX    65EB          NBLKB   0000          NBLKM   0001          NEWBOT  6810
NL      000A          NSADDR  6802          NUMBUF  64A3          NXTABC  0000:19 X
NXTLNK  0000:1A X     NXTPTR  0000:1B X     NXTVAL  0000:1C X     OLDCHR  64A0
OLDCUR  649F          OLDKEY  649D          OLDXY   65EF          ONEBUF  6729
OPRL    0014          OPRSP   655B          OPRSTK  6547          OPRSZ   655D
OUTOFF  62E7          PCNT    655E          PIXVAL  65ED          POINTE  64A7
PONOFF  65DA          PRINTR  6540          PUTLPT  0000:1E X     PUTOPN  0000:1D X
RAMEND  7FFF          RAMSTR  6900          RANSHT  655F          RMDTMP  6538
RUBACK  0005          RUBOUT  007F          SAVESP  625C          SCNEED  0004
SCRWIN  67CD          SDEL    0000:1F X     SNAP    0000:20 X     SOPRSP  653D
```

DOL - DOL--DOLOOP STUFF AND MIB STUFF
+++++ SYMBOL TABLE +++++

```
SOPRSZ 653F        STACK  60C8        STAKTO 6000        STOP   005D'   I
STRSIZ 6800        SUBSTU 6534        TAB    0009        TAPBUF 64B3
TAPCON 64AF        TAPPRO 64B1        TBFEND 6533        TEMPHD 60CA
TEMPS  62E5        TMPARG 67AD        TOP    6818        TTYBEG 6261
TTYEND 62E1        TTYINT 62E3        TTYPTR 62E1        TXTWIN 67E2
UARTFL 649C        USREND 65F1        V3PTR  6573        VDCHAR 649E
VDNLF  65CF        VIPLH  6545        VOICE0 6563        WRMODE 65EE
ZGIM2  0001        ZGREND 681D        $AADR  0010        $ADDRF 0007
$ADDRI 0005        $ADDRS 0009        $ANY   001A        $ANYNA FFFC
$ANYVA FFFE        $ARGPT 0011        $BGPTR 000F        $BNDL  0007
$CALLE 000D        $CMDAD 0018        $CPLBL 002A        $CSBLO 0028
$DATAP 0007        $DOLDE 0001        $DOLOO 0002        $DVAL  0000
$END   001C        $FADR  000E        $FLAGS 0002        $FORBL 0024
$FORPT 000B        $FVAL  0006        $GOSUB 001A        $IADR  000C
$INPBU 0018        $INPPT 0016        $IVAL  0004        $KEYBL 0026
$LENGT 0001        $LINPT 0005        $LOCPT 0009        $MIBBL 0022
$MIBEN 001B        $NAMAD 000A        $NAME  000A        $NASCI 0009
$NDEL  0080        $NLINK 0003        $NULL  0002        $NVALU 0005
$REPEA 001E        $RVSTU 0013        $SAME  0020        $SASCI 000A
$SLEN  0006        $STRAD 0008        $STRPT 0003        $TAF   0000
$TYPE  0000        $USE   0005        .BLNK. 0000:03 X   .DATA. 0000*   X
.PROG. 0153'   X
```

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                      ***
@                    *******
@                         *
@                        *
@                       *
@                      *
@                     *
@                    *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                 THE VERY BEST"
@
@              ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                    (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
          .LINK
          .PREL
          .IDENT  CM1
          .INTERN CMDTAB
          .INTERN CONTAB
          .INTERN DATA
          .INTERN END
          .INTERN FOR
          .INTERN GOTO
          .INTERN IF
          .INTERN LET
          .INTERN NEXT
          .INTERN RESTORE
          .INTERN RETURN
          .INTERN SKIP
          .INTERN THEN
          .EXTERN ABORT
          .EXTERN ACOPY
          .EXTERN ALLOC
          .EXTERN ALLOCD
          .EXTERN ARRAY
          .EXTERN ARG
          .EXTERN BOX
          .EXTERN CARTYP
          .EXTERN CLEARIT
          .EXTERN CMDREN
          .EXTERN CMDRES
          .EXTERN CMDSKP
          .EXTERN CORE
          .EXTERN DEC5IY
          .EXTERN DELETE
          .EXTERN DISPLAY
          .EXTERN DOMORE
```

```
.EXTERN ERRPGM
.EXTERN EVAL
.EXTERN EVALARG
.EXTERN EDIT
.EXTERN FINDHASH
.EXTERN FINDLAB
.EXTERN FORMAT
.EXTERN FREE
.EXTERN GETDSK
.EXTERN GETLPT
.EXTERN GETOPND
.EXTERN HELP
.EXTERN HEXHL
.EXTERN INC5IY
.EXTERN INIVDA
.EXTERN INIVDM
.EXTERN INIDEV
.EXTERN INPTTY
.EXTERN INPUT
.EXTERN ISNAME
.EXTERN ISNL
.EXTERN KISNL
.EXTERN KLB
.EXTERN LARGE
.EXTERN LINE
.EXTERN LINEP
.EXTERN MODIFY
.EXTERN NAMADR
.EXTERN NAMSET
.EXTERN NLPNT
.EXTERN NXTABC
.EXTERN NXTLNK
.EXTERN NXTPTR
.EXTERN NXTVAL
.EXTERN OUTCH
.EXTERN PDECBC
.EXTERN PRINT
.EXTERN PRINTF
.EXTERN PRINTHL
.EXTERN PRTYPA
.EXTERN PSCAN
.EXTERN PUTLPT
.EXTERN PUTOPND
.EXTERN RENAME
.EXTERN RETNONE
.EXTERN RETONE
.EXTERN RGBINI
.EXTERN SCANB
.EXTERN SCANLAB
.EXTERN SCANNL
.EXTERN SCLEAR
.EXTERN SMALL
.EXTERN SNAP
.EXTERN STOP
.EXTERN STREQK
```

```
                                .EXTERN SWITCH
                                .EXTERN TABPNT
                                .EXTERN TANGENT
                                .EXTERN TEXT
                                .EXTERN USEMAP
                                .EXTERN WHATSIS


0000'   0036'           CMDTAB: .WORD ANYARGS
0002'   0000:0A                .WORD BOX
0004'   0058'                  .WORD CONTROL
0006'   0445'                  .WORD DATA
0008'   0000:18                .WORD EDIT

000A'   0127'                  .WORD FOR
000C'   03F9'                  .WORD GOTO
000E'   0000:20                .WORD HELP
0010'   03B3'                  .WORD IF
0012'   637C                    .WORD AFIX[J][("J"-101Q)*16+FSTINT]
0014'   638C                    .WORD AFIX[K][("K"-101Q)*16+FSTINT]
0016'   0468'                  .WORD LET
0018'   0000:2F                .WORD MODIFY
001A'   0269'                  .WORD NEXT
001C'   02C4'                  .WORD ONERROR
001E'   0000:39                .WORD PRINT
0020'   63EC                    .WORD AFIX[Q][("Q"-101Q)*16+FSTINT]
0022'   030B'                  .WORD RETURN
0024'   036F'                  .WORD SKIP
0026'   0000:4F                .WORD TEXT
0028'   0000:50                .WORD USEMAP
002A'   643C                    .WORD AFIX[V][("V"-101Q)*16+FSTINT]
002C'   0000:51                .WORD WHATSIS
002E'   645C                    .WORD AFIX[X][("X"-101Q)*16+FSTINT]
0030'   646C                    .WORD AFIX[Y][("Y"-101Q)*16+FSTINT]
0032'   647C                    .WORD AFIX[Z][("Z"-101Q)*16+FSTINT]
0034'   648C                   FSTDOL

                        ; *ANYARGS COMMAND**

                        ;RETURNS 1 IF THERE'S ARGS LEFT, 0 OTHERWISE

                        M.CMD[ANYARGS,..ANYX,0,ARRAY,0,..ANY1][
0036'   18             +ANYARGS:        .BYTE $CMDADR
0037'   03             +        .BYTE (..ANYX-ANYARGS)/16+1
0038'   00             +        .BYTE 0
0039'   0000:08        +        .WORD ARRAY
003B'   0048'          +        .WORD ..ANY1
003D'   0000           +        .WORD 0
003F'   414E59415247+  +        .ASCIZ /ANYARGS/
                       +]
0047'   00                      .BYTE 0 ;NO SWITCHES
0048'   DD2A 625E     ..ANY1: LIXD MACTOP     ;SEE IF $MIBEND
004C'   DD7E1B                 MOV A,$MIBEND(X)        ;IS 0
004F'   B7                     ORA A           ;NOPE, RET 1
0050'   C0                     RNZ             ;RET TO RETONE
```

```
0051'   5F                      MOV E,A
0052'   57                      MOV D,A
0053'   3E04                    MVI A,$IVAL       ;RET A 0
0055'   C3 0000:41              JMP RETNONE
0058'                   ..ANYX:


                        ; ***CONTROL COMMAND***
                        M.CMD[CONTROL,CONTRX,0,CORE,CONTRH,CONTR1][
0058'   18             +CONTROL:       .BYTE $CMDADR
0059'   0D              +       .BYTE (CONTRX-CONTROL)/16+1
005A'   00              +       .BYTE 0
005B'   0000:10         +       .WORD CORE
005D'   006A'           +       .WORD CONTR1
005F'   006D'           +       .WORD CONTRH
0061'   434F4E54524F+    +       .ASCIZ /CONTROL/
                        +]
0069'   00                      .BYTE 0
006A'   CD 0000:09      CONTR1: CALL ARG
006D'   0800            CONTRH: .BYTE $STRADR,$TAF
006F'   E5                      PUSH H
0070'   FD6E01                  MOV L,1(Y)        ;GET ADDR
0073'   FD6602                  MOV H,2(Y)
0076'   01 000A                 LXI B,$SASCII     ;GET FIRST CHAR
0079'   09                      DAD B
007A'   7E                      MOV A,M
007B'   D640                    SUI 'A'-1
007D'   F2 0084'                JP ..OK           ;IS LT 0?
0080'                   ..BAD:  ERROR ER.CTL      ;BAD CHAR[
0080'   CD 0000:15      +       CALL ERRPGM
0083'   36              +       .BYTE ER.CTL
                        +]
0084'   FE1B             ..OK:   CPI 27  ;GT Z?
0086'   F2 0080'                 JP ..BAD          ;YUP
0089'   57              CONTRL: MOV D,A           ;SAVE A
008A'   87                      ADD A             ;DOUBLE IT
008B'   4F                      MOV C,A           ;ENTRY FROM IO SCANNER
008C'   0600                    MVI B,0
008E'   21 00AA'                LXI H,CONTAB-2    ;GO THRU TABLE
0091'   CD 0000:33              CALL NXTABC       ;GET VALUE
0094'   E5                      PUSH H            ;SAVE IT
0095'   21 00A3'                LXI H,CONRET      ;RET ADDR
0098'   E3                      XTHL              ;ON STACK
0099'   E5                      PUSH H            ;NOW FETCH FLAGS BYTE
009A'   21 65CA                 LXI H,FLAGS-1     ;HAS 26 BYTES IN TABLE
009D'   4A                      MOV C,D           ;GET CNTL CHAR BACK
009E'   09                      DAD B             ;B ZEROED BY MVI ABOVE
009F'   7E                      MOV A,M           ;GET FLAG VALUE
00A0'   EB                      XCHG              ;INTO DE
00A1'   E1                      POP H             ;GET GOTO ADDR
00A2'   E9                      PCHL              ;GOTOIT
00A3'   5F              CONRET: MOV E,A           ;SETUP FOR RETURN
00A4'   1600                    MVI D,0
00A6'   3E04                    MVI A,$IVAL
00A8'   E1                      POP H             ;GET H BACK (EOL PTR)
```

```
00A9'  C3 0000:41          JMP RETNONE      ;RETURN WITH A&DE SET

00AC'  00FD'       CONTAB: .WORD CXOR       ;A-AUDIO ON/OFF
00AE'  00EA'               .WORD CLARGE     ;B-BIG CHARS
00B0'  00F6'               .WORD CSINC      ;CONTROL C
00B2'  00FD'               .WORD CXOR       ;D-DEBUG/SSTEP OFF/ON
                                            ;CLEARED BY ABORT
00B4'  00FD'               .WORD CXOR       ;E-ECHO ON/OFF
00B6'  00FD'               .WORD CXOR       ;F-VIP ON/OFF
00B8'  0101'               .WORD CLRALL     ;G-CLEAR ALL
00BA'  00FC'               .WORD CSET       ;H-EDITOR <-
00BC'  00FC'               .WORD CSET       ;I-EDITOR INSERT
00BE'  00FC'               .WORD CSET       ;J-EDITOR DOWNER
00C0'  00FC'               .WORD CSET       ;K-EDITOR UPPER
00C2'  00FC'               .WORD CSET       ;L-EDITOR ->
00C4'  00FD'               .WORD CXOR       ;M (RESERVED--CR)
00C6'  0115'               .WORD CNOISE     ;N-SHUT OFF MUSIC
00C8'  00FD'               .WORD CXOR       ;O-PRINT ON/OFF
                                            ;CLEARED BY ABORT
00CA'  00FC'               .WORD CSET       ;P-PRINT OFF
                                            ;CLEARED BY ABORT
00CC'  0123'               .WORD CQ         ;Q-PRINT ON
00CE'  0000:43             .WORD RGBINI     ;R-DEFAULT COLORS
00D0'  00F1'               .WORD CSMALL     ;S-CHARSET S/L
00D2'  00E5'               .WORD INITWO     ;T-TWO LINE WINDOW
00D4'  00FD'               .WORD CXOR       ;U-UNUSED
00D6'  00FC'               .WORD CSET       ;V-EDITOR
00D8'  00E0'               .WORD INIWHL     ;W-WHOLE WINDOW
00DA'  00FD'               .WORD CXOR       ;X-LIST
00DC'  00FC'               .WORD CSET       ;Y-EDITOR
00DE'  00FC'               .WORD CSET       ;Z-CONTRLZ
                                            ;CLEARED BY ABORT
                   ;NOW THE ACTION:
00E0'  21 0000:24  INIWHL: LXI H,INIVDM
00E3'  1808                JMPR CLARX
00E5'  21 0000:23  INITWO: LXI H,INIVDA
00E8'  1803                JMPR CLARX
00EA'  21 0000:2C  CLARGE: LXI H,LARGE
00ED'  22 6534     CLARX:  SHLD SUBSTUFF
00F0'  C9                  RET
00F1'  21 0000:48  CSMALL: LXI H,SMALL
00F4'  18F7                JMPR CLARX
00F6'  3C          CSINC:  INR A            ;INC UNTIL HITS 5
00F7'  FE05                CPI 5
00F9'  F2 0000:04          JP ABORT         ;BINGO
00FC'  AF          CSET:   XRA A            ;SET TO 1 BY CLEAR/XOR
00FD'  EE01        CXOR:   XRI 1            ;FLIP IT
00FF'  12                  STAX D           ;STUFF IT
0100'  C9                  RET
0101'  21 65CB     CLRALL: LXI H,FLAGS
0104'  01 001A             LXI B,26
0107'  CD 0000:0C          CALL CLEARIT
010A'  21 6593             LXI H,HCAREA     ;ZAP HAND CONTROLS
010D'  01 0010             LXI B,16
0110'  CD 0000:0C          CALL CLEARIT
```

```
0113'   AF                      XRA A
0114'   C9                      RET
0115'   AF              CNOISE: XRA A
0116'   32 65BD                 STA DEVVA
0119'   32 65C1                 STA DEVVB
011C'   32 65C5                 STA DEVVC
011F'   32 65B9                 STA DEVVN
0122'   C9                      RET
0123'   1B              CQ:     DCX D           ;ZAP P BYTE
0124'   AF              COUT:   XRA A
0125'   12                      STAX D          ;CLEAR IT
0126'   C9                      RET
0127'                   CONTRX:

                        ;***FOR COMMAND***
                        M.CMD[FOR,..FORX,0,FORMAT,..FORA,..FOR1][
0127'   18             +FOR:    .BYTE $CMDADR
0128'   13             +        .BYTE (..FORX-FOR)/16+1
0129'   00             +        .BYTE 0
012A'   0000:1B        +        .WORD FORMAT
012C'   0157'          +        .WORD ..FOR1
012E'   0135'          +        .WORD ..FORA
0130'   464F5200       +        .ASCIZ /FOR/
                       +]
0134'   00                      .BYTE 0         ;NO SWITCHES
0135'   63              ..FORA: .BYTE 99        ;SPECIAL
0136'   4142433D3120            .ASCIZ /ABC=1 TO 10 STEP 2 (FOR EXAMPLE)/
                        ;FORBLK SETUP:
                        ;0      $TYPE
                        ;1      $LENGTH
                        ;2      $FLAGS
                        ;3&4    PTR TO PREV FORBLK OR TO MIB
                        ;5&6    NOT USED
                        ;7&8    ADDR OF CODE FOLLOWING FOR STATEMENT
                        ;9....  ASCII REPRESENTING INCREMENT AND
                        ;       CONDITIONAL STATEMENTS CREATED BY FOR

                        ;       GENERAL MODE OF ATTACK IS TO:
                        ;1. ALLOCATE FORBLK AND CHAIN IN AT $FORPTR
                        ;4. IF IT'S VAR= CHANGE TO VAR_
                        ;5. EVALUATE VAR_EXPR1
                        ;6. IGNORE "TO" IF THERE
                        ;7. SAVE PTR TO EXPR2
                        ;8. SCAN OVER EXPR2 & SCAN OVER "STEP" IF THERE
                        ;9. IF NO EXPR3, MAKE EXPR3=1
                        ;10. CHECK SIGN OF EXPR3. IF NEG, CHANGE > TO <
                        ;11. CREATE STRING VAR_VAR+EXPR3
                        ;12. CREATE STRING VAR>EXPR2 OR VAR<EXPR2

0157'   CD 0000:2B     ..FOR1: CALL KLB        ;ZIP PAST BLANX
015A'   E5                      PUSH H          ;SAVE PTR TO VAR FOR EVAL
015B'   CD 0000:31              CALL NAMSET     ;POINT TO DELIM
015E'   FE28                    CPI '('         ;IS ARRAY?
0160'   CC 0000:3D             CZ PSCAN        ;YEAH, CHECK PAST )
0163'   365F                    MVI M,AASN      ;STUFF A AASN IN
```

```
0165'   3E05                    MVI A,5         ;GET 6 BLOX FER NOW
0167'   CD 0000:06              CALL ALLOC
016A'   E5                      PUSH H          ;IX<- FWA OF FORBLK
016B'   DDE1                    POP X
016D'   3624                    MVI M,$FORBLK
016F'   2A 625E                 LHLD MACTOP     ;CHAIN IT IN
0172'   01 000B                 LXI B,$FORPTR   ;GET PREV FORBLK
0175'   09                      DAD B
0176'   7E                      MOV A,M
0177'   DD7703                  MOV 3(X),A      ;PUT IN NEW BLOCK
017A'   23                      INX H
017B'   7E                      MOV A,M
017C'   DD7704                  MOV 4(X),A      ;LIKE THIS
017F'   DDE5                    PUSH X
0181'   D1                      POP D           ;NEW FORBLK PTR IN DE
0182'   72                      MOV M,D         ;PUT IN PREV FORBLK
0183'   2B                      DCX H
0184'   73                      MOV M,E         ;LIKE THIS
0185'   E1                      POP H           ;PTR TO VAR
0186'   E5                      PUSH H          ;SAVE IT
0187'   CD 0000:17              CALL EVALARG    ;DO ASSIGNMENT
018A'   CD 0000:2A              CALL KISNL      ;ANY MORE
018D'   CA 0247'                JZ ..FORE       ;NOPE-ERROR
0190'   FE2C                    CPI ',          ;COMMA?
0192'   2003                    JRNZ ..FOR2
0194'   23                      INX H           ;JUMP OVER IT
0195'   180D                    JMPR ..NOTO
0197'   11 01D6'        ..FOR2: LXI D,..TOA     ;SKIP OVER "TO"
019A'   CD 0000:4B              CALL STREQK
019D'   2005                    JRNZ ..NOTO     ;NOT THERE
019F'   CD 0000:44              CALL SCANB      ;SKIP TO BLANK
01A2'   2847                    JRZ ..FORJ      ;IF EOL, ERROR
01A4'   CD 0000:2B      ..NOTO: CALL KLB
01A7'   E3                      XTHL            ;SP->EXPR2, HL->VAR
01A8'   E5                      PUSH H          ;SAVE ->VAR
01A9'   DDE5                    PUSH X          ;POINT DE->FORASCII
01AB'   D1                      POP D           ;...
01AC'   7B                      MOV A,E         ;ADD 9 TO IT
01AD'   C609                    ADI 9           ;CLUMSY
01AF'   5F                      MOV E,A
01B0'   AF                      XRA A
01B1'   8A                      ADC D
01B2'   57                      MOV D,A         ;WHEW!
01B3'   3E5F                    MVI A,'_'       ;COPY TILL _
01B5'   CD 0000:05              CALL ACOPY
01B8'   E1                      POP H
01B9'   CD 0000:05              CALL ACOPY      ;VAR_VAR_
01BC'   1B                      DCX D           ;VAR_VAR
01BD'   E1                      POP H           ;H->EXPR
01BE'   E5                      PUSH H          ;AND SAVE IT AGAIN
01BF'   3E2B                    MVI A,'+'       ;BUILD VAR_VAR+1
01C1'   12                      STAX D
01C2'   13                      INX D
01C3'   CD 0000:44              CALL SCANB      ;SCAN OVER
01C6'   2011                    JRNZ ..STEP     ;SUMPTIN THERE
```

```
01C8'   3E31                MVI A,'1'
01CA'   12                  STAX D
01CB'   13                  INX D
01CC'   AF                  XRA A
01CD'   12                  STAX D  ;PUT IN TERMINATOR
01CE'   13                  INX D
01CF'   1839                JMPR ..PLUS
01D1'   5354455000  ..STEA: .ASCIZ /STEP/
01D6'   544F00      ..TOA:  .ASCIZ /TO/
01D9'   CD 0000:2B  ..STEP: CALL KLB
01DC'   D5                  PUSH D          ;SCAN OVER "STEP" IF THERE
01DD'   11 01D1'            LXI D,..STEA
01E0'   CD 0000:4B          CALL STREQK
01E3'   2009                JRNZ ..NOST     ;NOSTEP!
01E5'   CD 0000:45          CALL SCANLAB    ;SKIP OVER
01E8'   CD 0000:29          CALL ISNL       ;CHECK TERMY
01EB'   CA 0247'    ..FORJ: JZ ..FORE       ;NUTTIN FOLLOWING "STEP"
01EE'   D1          ..NOST: POP D           ;BACK INTO FORBLK
01EF'   CD 0000:2B          CALL KLB
01F2'   FE2C                CPI ','         ;COMMA?
01F4'   2001                JRNZ ..NOSE
01F6'   23                  INX H
01F7'   E5          ..NOSE: PUSH H          ;SVE PTR TO EXPR3
01F8'   EDA0        ..SORE: LDI             ;COPY TILL EOL
01FA'   CD 0000:29          CALL ISNL
01FD'   20F9                JRNZ ..SORE
01FF'   EDA0                LDI             ;COPY TERMY TOO
0201'   E1                  POP H           ;PTR TO EXPR 3
0202'   3E04                MVI A,$IVAL     ;NEED INTEGER
0204'   CD 0000:16          CALL_EVAL       ;GET SIGN OF EXPR3
0207'   FD7E02              MOV A,E.HVAL(Y)         ;INTO A
020A'   E5          ..PLUS: PUSH H          ;SAVE EOL FOR CMDRET
020B'   F5                  PUSH PSW        ;SAVE SIGN IN A
020C'   DDE5                PUSH X          ;GET FORBLK FWA IN HL
020E'   E1                  POP H
020F'   01 0009             LXI B,9
0212'   09                  DAD B           ;GET PTR TO VAR IN FORBLK
0213'   3E5F                MVI A,'_'
0215'   CD 0000:05          CALL ACOPY
0218'   1B                  DCX D           ;IGNORE TERMINATOR
0219'   EB                  XCHG            ;GET HL FROM DE
021A'   363E                MVI M,'>'       ;DEFAULT
021C'   F1                  POP PSW
021D'   B7                  ORA A           ;CHECK SIGN
021E'   F2 0223'            JP ..SOK        ;IS POSITIVE
0221'   363C                MVI M,'<'       ;IS NEGATIVE
0223'   23          ..SOK:  INX H
0224'   EB                  XCHG            ;DE=HL
0225'   E1                  POP H           ;EOL BACK
0226'   E3                  XTHL            ;ON STACK, HL->EXPR2
0227'   EDA0        ..COPY: LDI             ;COPY TILL BLANK OR NL
0229'   7E                  MOV A,M
022A'   FE20                CPI ' '
022C'   2809                JRZ ..E2X       ;BUILD VAR<EXPR2
022E'   FE2C                CPI ','
```

```
0230'   2805                    JRZ ..E2X
0232'   CD 0000:29              CALL ISNL
0235'   20F0                    JRNZ ..COPY
0237'   EDA0          ..E2X:    LDI                 ;COPY DELIM TOO
0239'   DDE5                    PUSH X              ;GET FORBLOK FWA
023B'   E1                      POP H               ;FOR ALLOCD
023C'   CD 0000:07              CALL ALLOCD         ;DE->1 PAST
023F'   E1                      POP H               ;EOL
0240'   DD7507                  MOV 7(X),L
0243'   DD7408                  MOV 8(X),H          ;FOR NEXT
0246'   C9                      RET
0247'   E5          ..FORE:   PUSH H
0248'   DDE5                    PUSH X
024A'   E1                      POP H
024B'   CD 0253'                CALL FORDEL
024E'   E1                      POP H
                                ERROR ER.FOR[
024F'   CD 0000:15      +       CALL ERRPGM
0252'   28              +       .BYTE ER.FOR
                        +]
0253'                    ..FORX:
0253'   E5              FORDEL: PUSH H  ;SAVE FORBLK PTR FOR FREE
0254'   DDE5                    PUSH X
0256'   CD 0000:34              CALL NXTLNK         ;GET PREV FORBLK
0259'   DD2A 625E                LIXD MACTOP        ;CURRENT MIB
025D'   DD750B                  MOV $FORPTR(X),L
0260'   DD740C                  MOV $FORPTR+1(X),H
0263'   DDE1                    POP X
0265'   E1                      POP H
0266'   C3 0000:1C              JMP FREE


                         M.CMD[NEXT,..NEXX,0,AFIX[N],..NEXA,..NEX1][
0269'   18              +NEXT:    .BYTE $CMDADR
026A'   06              +         .BYTE (..NEXX-NEXT)/16+1
026B'   00              +         .BYTE 0
026C'   63BC            +         .WORD AFIX[N][("N"-101Q)*16+FSTINT]
026E'   0278'           +         .WORD ..NEX1
0270'   027B'           +         .WORD ..NEXA
0272'   4E45585400      +         .ASCIZ /NEXT/
                        +]
0277'   00                      .BYTE 0             ;NO SWITCHES
0278'   CD 0000:09     ..NEX1:  CALL ARG            ;GET VAR ADDR
027B'   0A00            ..NEXA:  .BYTE $NAME,$TAF
027D'   E5                      PUSH H
027E'   2A 625E                 LHLD MACTOP         ;GET AT FORBLK
0281'   01 000B                 LXI B,$FORPTR       ;LATEST ENTRY
0284'   CD 0000:33              CALL NXTABC         ;IF ZERO, WE'RE DEAD
0287'                    ..NXTE:  ZERROR ER.NXT[
0287'   2004            +         JRNZ ..0001
0289'   CD 0000:15      +         CALL ERRPGM
028C'   1D              +         .BYTE ER.NXT
028D'                    +..0001:]
028D'   E5                      PUSH H              ;HL HAS FORBLK ADDR
028E'   01 0009                 LXI B,9             ;FIRST DO INCR EVAL
```

```
0291'   09                      DAD B
0292'   CD 0000:17              CALL EVALARG     ;IGNORE RESULTS
0295'   23                      INX H            ;POP OVER TERMY
0296'   3E04                    MVI     A,$IVAL  ;ASK FOR INTEGER
0298'   CD 0000:16              CALL EVAL        ;CHECK OUT CONDITIONAL
029B'   FD7E01                  MOV A,E.LVAL(Y)          ;GET VALUE
029E'   E1                      POP H
029F'   B7                      ORA A            ;IS 0 (FAIL, I.E. GO UP)?
02A0'   200A                    JRNZ ..FALL      ;NOPE, FALL THRU, DEL FORBLK
02A2'   01 0007                 LXI B,7          ;GET PTR TO BACK ADDR
02A5'   CD 0000:33              CALL NXTABC      ;GET IT
02A8'   28DD                    JRZ ..NXTE       ;BAD ERROR
02AA'   F1                      POP PSW          ;TRASH OLD HL EOL
02AB'   C9                      RET              ;JUST RETURN

02AC'   E5              ..FALL: PUSH H           ;SAVE CURRENT FORBLK
02AD'   DDE5                    PUSH X
02AF'   CD 0000:34              CALL NXTLNK      ;POINT TO PREV FORBLOK
02B2'   DD2A 625E               LIXD MACTOP
02B6'   DD750B                  MOV $FORPTR(X),L
02B9'   DD740C                  MOV $FORPTR+1(X),H
02BC'   DDE1                    POP X
02BE'   E1                      POP H
02BF'   CD 0000:1C              CALL FREE        ;ZAP CURRENT
02C2'   E1                      POP H            ;EOL PTR
02C3'   C9                      RET              ;FALL THRU
02C4'                    ..NEXX:
                         M.CMD[ONERROR,..ONEX,0,AFIX[0],..ONEA,..ONE1][
02C4'   18              +ONERROR:       .BYTE $CMDADR
02C5'   05              +       .BYTE (..ONEX-ONERROR)/16+1
02C6'   00              +       .BYTE 0
02C7'   63CC            +       .WORD AFIX[0][("0"-101Q)*16+FSTINT]
02C9'   02E7'           +       .WORD ..ONE1
02CB'   02D6'           +       .WORD ..ONEA
02CD'   4F4E4552524F+   +       .ASCIZ /ONERROR/
                        +]
02D5'   00                      .BYTE 0          ;NO SWITCHES
02D6'   63              ..ONEA: .BYTE 99
02D7'   414E59205A47            .ASCIZ /ANY ZGRASS CODE/

02E7'   01 0000         ..ONE1: LXI B,0          ;CLEAR OUT ONEBUF A BIT
02EA'   ED43 6729               SBCD ONEBUF
02EE'   ED43 672B               SBCD ONEBUF+2
02F2'   CD 0000:2A              CALL KISNL       ;EOL?
02F5'   C8                      RZ               ;YUP, RETURN
02F6'   11 672D                 LXI D,ONEBUF+4   ;START AT #4
02F9'   EDA0            ..MORE: LDI              ;COPY TILL DONE
02FB'   CD 0000:29              CALL ISNL        ;...
02FE'   20F9                    JRNZ ..MORE      ;...
0300'   FE3B                    CPI ';'          ;COPY ';' TOO
0302'   28F5                    JRZ ..MORE
0304'   AF                      XRA A            ;STUFF A NULL IN
0305'   12                      STAX D           ;AT END
0306'   3C                      INR A            ;AND SET ONEBUF FIRST
0307'   32 6729                 STA ONEBUF
```

```
030A'  C9                      RET
030B'                  ..ONEX:

                       M.CMD[RETURN,RETX,0,REMARK,..RETA,..RET1][
030B'  18              +RETURN: .BYTE $CMDADR
030C'  04              +        .BYTE (RETX-RETURN)/16+1
030D'  00              +        .BYTE 0
030E'  0454'           +        .WORD REMARK
0310'  031C'           +        .WORD ..RET1
0312'  031F'           +        .WORD ..RETA
0314'  52455455524E+   +        .ASCIZ /RETURN/
                       +]
031B'  00                       .BYTE 0          ;NO SWITCHES
031C'  CD 0000:09      ..RET1:  CALL    ARG
031F'  FE1E            ..RETA:  .BYTE   $ANYVAL,$REPEAT
0321'  CD 0000:1F               CALL    GETOPND ;GET THE OPERAND
0324'  B7                       ORA A           ;IS 0?
0325'  2004                    JRNZ ..RETB     ;NO-GET ARG
0327'  21 033D'                LXI H,RFAKE     ;FAKE A NULL
032A'  C9                       RET             ;BACK TO RETONE
032B'  FD2A 625E       ..RETB:  LIYD    MACTOP  ;IY -> MIB
032F'  01 0013                  LXI B,$RVSTUFF  ;SETUP Y
0332'  FD09                     DADY B
0334'  CD 0000:3F               CALL PUTOPND    ;STUFF IT
0337'  21 033D'        RETOUT:  LXI H,RFAKE     ;STUFF IT
033A'  C3 0000:41               JMP RETNONE     ;AND RETURN VALUE
033D'                  RETX:
033D'  0000            RFAKE:   .WORD 0 ;FAKE NULL EOF


033F'                  ..RESX:
                       M.CMD[GOSUB,..GOSX,0,GETDSK,GOTARG,..GOS1][
033F'  18              +GOSUB:  .BYTE $CMDADR
0340'  04              +        .BYTE (..GOSX-GOSUB)/16+1
0341'  00              +        .BYTE 0
0342'  0000:1D         +        .WORD GETDSK
0344'  034F'           +        .WORD ..GOS1
0346'  0408'           +        .WORD GOTARG
0348'  474F53554200+   +        .ASCIZ /GOSUB/
                       +]
034E'  00                       .BYTE 0          ;NO SWITCHES
034F'  CD 0000:2B      ..GOS1:  CALL KLB         ;START LIKE GOTO
0352'  DDE5                     PUSH X          ;PUTSTUFF ON STACK
0354'  FDE5                     PUSH Y          ;...
0356'  E5                       PUSH H          ;SAVE ->TO LABEL
0357'  CD 0000:46               CALL SCANNL     ;GET LINE TO RET TO
035A'  E3                       XTHL            ;REPLACE ON STACK
035B'  11 0000:42               LXI D,RETONE    ;RETURN ADDR
035E'  D5                       PUSH D
035F'  FD2A 625E                LIYD MACTOP     ;GET &INCR GOSUB BYTE
0363'  FD341A                   INR $GOSUB(Y)   ;TO INDICATE GOSUB
0366'  CD 0000:1A               CALL FINDLAB    ;GET TO CODE
0369'  CD 0000:3E               CALL PUTLPT     ;SAVE LINE PTR
036C'  C3 0000:14               JMP DOMORE      ;AND EXECUTE IT
036F'                  ..GOSX:
```

```
                        M.CMD[SKIP,..SKPX,0,STOP,..SKPA,..SKP1][
036F'   18              +SKIP:    .BYTE $CMDADR
0370'   05              +         .BYTE (..SKPX-SKIP)/16+1
0371'   00              +         .BYTE 0
0372'   0000:4A         +         .WORD STOP
0374'   037E'           +         .WORD ..SKP1
0376'   0381'           +         .WORD ..SKPA
0378'   534B495000      +         .ASCIZ /SKIP/
                        +]
037D'   00                        .BYTE 0           ;NO SWITCHES
037E'   CD 0000:09      ..SKP1:   CALL ARG
0381'   0400            ..SKPA:   .BYTE $IVAL,0     ;GET A NUMBER
0383'   FD5E01                    MOV E,1(Y)
0386'   FD5602                    MOV D,2(Y)        ;IN DE
0389'   AF                        XRA A             ;CHECK FOR 0
038A'   B2                        ORA D             ;IS +, 0 OR -?
038B'   FA 039D'                  JM ..SKP          ;ITS NEGATIVE
038E'   B3                        ORA E             ;IS ALL ZERO?
038F'   280C                      JRZ ..SKP         ;ITS  0
0391'   CD 0000:46     ..SKDN:   CALL SCANNL       ;SKIP NL'S UNTIL DE=0
0394'   1B                        DCX D
0395'   B7                        ORA A             ;CHECK FOR NULL
0396'   C8                        RZ                ;IS NULL (FROM SCANNL)
0397'   7A                        MOV A,D           ;CHECK FOR DE=0
0398'   B3                        ORA E
0399'   20F6                      JRNZ ..SKDN       ;...
039B'   2B                        DCX H             ;BACKUP&->NL
039C'   C9                        RET               ;ALL DONE
039D'   2B              ..SKP:    DCX H             ;BACKUP ONE
039E'   7E                        MOV A,M           ;PICKUP CURRENT CHAR
039F'   B7                        ORA A
03A0'   2004                      JRNZ ..NLCK       ;IS NO NULL
03A2'   23                        INX H             ;IS NULL, POINT AT FIRST
03A3'   C3 0000:0E                JMP CMDRES        ;CHAR OF STRING
03A6'   FE0A            ..NLCK:   CPI NL            ;IS NL?
03A8'   20F3                      JRNZ ..SKP        ;NOPE
03AA'   7A                        MOV A,D           ;CHECK IF DE=0
03AB'   B3                        ORA E             ;
03AC'   13                        INX D             ;INCR JUST IN CASE
03AD'   20EE                      JRNZ ..SKP        ;MORE
03AF'   C9                        RET               ;NO MORE
03B0'   2B              ..DCX:    DCX H             ;DE=DE-1
03B1'   18EA                      JMPR ..SKP        ;ZIP MORE
03B3'                   ..SKPX:

                                  M.CMD[IF,..FXX,0,INPUT,..IFA,..FI1][
03B3'   18              +IF:      .BYTE $CMDADR
03B4'   05              +         .BYTE (..FXX-IF)/16+1
03B5'   00              +         .BYTE 0
03B6'   0000:27         +         .WORD INPUT
03B8'   03C0'           +         .WORD ..FI1
03BA'   03DC'           +         .WORD ..IFA
03BC'   494600          +         .ASCIZ /IF/
                        +]
```

```
CM1     -
IF COMMAND


03BF'   00                      .BYTE 0         ;NO SWITCHES
03C0'   CD 0000:2B      ..FI1:  CALL KLB        ;JUMP OVER BLANX
                        ; ** CHANGE **
03C3'   CD 0000:22              CALL    INC5IY
03C6'   3E04                    MVI A,$IVAL     ;ASK FOR INTEGER
03C8'   CD 0000:16              CALL EVAL       ;0 IF FALSE, 1 IF TRUE
03CB'   FD7E01                  MOV A,E.LVAL(Y)         ;GET VALUE
03CE'   FDB602                  ORA E.HVAL(Y)           ;CHECK VALUE
03D1'   F5                      PUSH    PSW
03D2'   CD 0000:11              CALL    DEC5IY
03D5'   F1                      POP     PSW
03D6'   CA 0000:0F              JZ CMDSKP       ;FERGET IT
03D9'   C3 0486'                JMP THEN1       ;DO REST OF LINE
03DC'   63              ..IFA:  .BYTE 99        ;SPECIAL FOLLOWS
03DD'   413C31302C47            .ASCIZ /A<10,GOTO 123 (FOR EXAMPLE)/
03F9'                   ..FXX:


                        M.CMD[GOTO,GOTOX,0,GOSUB,GOTARG,GOTO1][
03F9'   18              +GOTO:  .BYTE $CMDADR
03FA'   02              +       .BYTE (GOTOX-GOTO)/16+1
03FB'   00              +       .BYTE 0
03FC'   033F'           +       .WORD GOSUB
03FE'   040F'           +       .WORD GOTO1
0400'   0408'           +       .WORD GOTARG
0402'   474F544F00      +       .ASCIZ /GOTO/
                        +]
0407'   00                      .BYTE 0         ;NO SWITCHES
0408'   63              GOTARG: .BYTE 99
0409'   4C4142454C00            .ASCIZ /LABEL/
040F'   CD 0000:2B      GOTO1:  CALL KLB        ;ZAP BLANX
0412'   CD 0000:1A              CALL FINDLAB    ;GET PTR TO LABEL
0415'   C3 0000:3E              JMP PUTLPT      ;AND STORE IT
0418'                   GOTOX:                  ;AND RETURN



                                M.CMD[RESTORE,..RESX,0,AFIX[R],0,..RES1][
0418'   18              +RESTORE:       .BYTE $CMDADR
0419'   02              +       .BYTE (..RESX-RESTORE)/16+1
041A'   00              +       .BYTE 0
041B'   63FC            +       .WORD AFIX[R][("R"-101Q)*16+FSTINT]
041D'   042A'           +       .WORD ..RES1
041F'   0000            +       .WORD 0
0421'   524553544F52+           .ASCIZ /RESTORE/
                        +]
0429'   00                      .BYTE 0         ;NO SWITCHES
042A'   FD2A 625E       ..RES1: LIYD MACTOP
042E'   FD360700                MVI $DATAPTR(Y),0
0432'   FD360800                MVI $DATAPTR+1(Y),0     ;CLEAR OUT
                                                        ;DATAPTR IN MIB
0436'   C9                      RET
0437'                   ..RESX:
```

```
                                    M.CMD[END,ENDX,0,AFIX[E],0,END1][
0437'   18              +END:       .BYTE $CMDADR
0438'   04              +           .BYTE (ENDX-END)/16+1
0439'   00              +           .BYTE 0
043A'   632C            +           .WORD AFIX[E][("E"-101Q)*16+FSTINT]
043C'   0465'          +           .WORD END1
043E'   0000            +           .WORD 0
0440'   454E4400        +           .ASCIZ /END/
                        +]
0444'   00                          .BYTE 0             ;NO SWITCHES
                                    M.CMD[DATA,ENDX,0,DELETE,0,END1][
0445'   18              +DATA:      .BYTE $CMDADR
0446'   03              +           .BYTE (ENDX-DATA)/16+1
0447'   00              +           .BYTE 0
0448'   0000:12         +           .WORD DELETE
044A'   0465'          +           .WORD END1
044C'   0000            +           .WORD 0
044E'   4441544100      +           .ASCIZ /DATA/
                        +]
0453'   00                          .BYTE 0             ;NO SWITCHES
                                    M.CMD[REMARK,ENDX,0,RENAME,0,END1][
0454'   18              +REMARK:    .BYTE $CMDADR
0455'   02              +           .BYTE (ENDX-REMARK)/16+1
0456'   00              +           .BYTE 0
0457'   0000:40         +           .WORD RENAME
0459'   0465'          +           .WORD END1
045B'   0000            +           .WORD 0
045D'   52454D41524B+   +           .ASCIZ /REMARK/
                        +]
0464'   00                          .BYTE 0             ;NO SWITCHES
0465'   C3 0000:0F      END1:       JMP CMDSKP
0468'                    ENDX:



                                    M.CMD[LET,..LETX,0,LINE,0,..LET1][
0468'   18              +LET:       .BYTE $CMDADR
0469'   01              +           .BYTE (..LETX-LET)/16+1
046A'   00              +           .BYTE 0
046B'   0000:2D         +           .WORD LINE
046D'   0476'          +           .WORD ..LET1
046F'   0000            +           .WORD 0
0471'   4C455400        +           .ASCIZ /LET/
                        +]
0475'   00                          .BYTE 0             ;NO SWITCHES
0476'   C9               ..LET1:    RET
0477'                    ..LETX:

                                    M.CMD[THEN,THENX,0,TANGENT,0,THEN1][
0477'   18              +THEN:      .BYTE $CMDADR
0478'   02              +           .BYTE (THENX-THEN)/16+1
0479'   00              +           .BYTE 0
047A'   0000:4E         +           .WORD TANGENT
```

CM1 -
THEN COMMAND

```
047C'  0486'        +          .WORD THEN1
047E'  0000         +          .WORD 0
0480'  5448454E00   +          .ASCIZ /THEN/
                    +]
0485'  00                      .BYTE 0          ;NO SWITCHES
0486'  E5           THEN1:     PUSH H
0487'  CD 0000:2B              CALL KLB         ;SCRUNCH BLANX
048A'  CD 0000:0B              CALL CARTYP      ;IF NUMERIC, ITS A LABEL
048D'  E1                      POP H            ;RESTORE H
048E'  DA 040F'                JC GOTO1
0491'  C9                      RET
0492'               THENX:


                               .END
```

CM1 -

+++++ SYMBOL TABLE +++++

```
AASN    005F          ABORT   0000:04 X     ACOPY   0000:05 X     ALLOC   0000:06 X
ALLOCD  0000:07 X     ANYARG  0036'         ARG     0000:09 X     ARGSTK  60CC
ARRAY   0000:08 X     BACKGR  65CC          BLANK   0020          BOTRAM  6000
BOTTOM  64A9          BOX     0000:0A X     CARTYP  0000:0B X     CHARSL  65DD
CLARGE  00EA'         CLARX   00ED'         CLEARI  0000:0C X     CLEARS  60CA
CLRALL  0101'         CMDREN  0000:0D X     CMDRES  0000:0E X     CMDSKP  0000:0F X
CMDTAB  0000'    I    CNOISE  0115'         CNTRL   000C          CNTRLC  65CD
CNTRLO  65D9          CNTRLU  0015          CNTRLZ  65E4          CONRET  00A3'
CONTAB  00AC'    I    CONTR1  006A'         CONTRH  006D'         CONTRL  0089'
CONTRO  0058'         CONTRX  0127'         CORE    0000:10 X     COUT    0124'
CPLARE  611C          CPLSIZ  0140          CQ      0123'         CR      000D
CSBLOK  668D          CSET    00FC'         CSFLAG  6260          CSINC   00F6'
CSMALL  00F1'         CURCX   6814          CURCY   6816          CURREN  64A5
CXOR    00FD'         C.CO    0011          C.C1    0012          C.CX    000B
C.CY    000D          C.DP    0013          C.ST    0002          C.X     0003
C.XF    000F          C.XS    0007          C.Y     0005          C.YF    0010
C.YS    0009          DATA    0445'    I    DDTON   65CE          DEC5IY  0000:11 X
DELETE  0000:12 X     DEVBL   6589          DEVCLO  6579          DEVCL1  657B
DEVCL2  657D          DEVCL3  657F          DEVCL4  6581          DEVCL5  6583
DEVCL6  6585          DEVCL7  6587          DEVFB   65CB          DEVHCB  658F
DEVMO   658B          DEVNM   65B7          DEVNT   658D          DEVTNA  65BB
DEVTNB  65BF          DEVTNC  65C3          DEVVA   65BD          DEVVAR  6579
DEVVB   65C1          DEVVBL  6591          DEVVC   65C5          DEVVD   65C9
DEVVN   65B9          DEVVS   65C7          DEVXCD  65B3          DEVYCD  65B5
DISPLA  0000:13 X     DOLPLH  62E8          DOLPPT  62EA          DOMORE  0000:14 X
DUMBST  6577          EDBCNT  64AD          EDCCNT  64A7          EDIT    0000:18 X
EDLONG  6812          EDMODE  681C          EDNAME  64A1          EDNCX   680C
EDNCY   680E          EDNEWS  64A5          EDOCX   6808          EDOCY   680A
EDPN    6806          EDPO    6804          EDPTRC  64AB          EDPTRL  64A9
EDSTR   681A          END     0437'    I    END1    0465'         ENDX    0468'
ERABIT  0002          ERRPGM  0000:15 X     ER.ARA  002F          ER.ARG  0034
ER.ASN  0015          ER.BOX  001A          ER.CHN  0002          ER.CMD  001F
ER.CNV  0016          ER.COR  001B          ER.CTL  0036          ER.DEL  0026
ER.DIM  0030          ER.DIV  0018          ER.DP   0037          ER.DSK  0019
ER.EDT  0035          ER.FMT  0038          ER.FNF  001C          ER.FOR  0028
ER.IMP  0003          ER.LAB  0025          ER.MAC  0022          ER.NAE  002D
ER.NAM  0029          ER.NEG  003A          ER.NOT  0023          ER.NUL  0039
ER.NUM  002B          ER.NXT  001D          ER.OFL  0017          ER.OPN  0014
ER.OVE  001E          ER.PAR  002A          ER.REN  002C          ER.RET  0024
ER.SEP  0021          ER.SNP  0031          ER.SPC  002E          ER.STK  0004
ER.SW   0032          ER.TER  0020          ER.UFL  0033          ER.UNF  0027
EVAL    0000:16 X     EVALAR  0000:17 X     EXTDEL  002E          E.HVAL  0002
E.LVAL  0001          E.SIZ   0005          E.TYP   0000          E.VAL   0001
FCNTH   65D2          FCNTI   65D3          FCNTJ   65D4          FCNTK   65D5
FCNTL   65D6          FCNTV   65E0          FCNTY   65E3          FINDHA  0000:19 X
FINDLA  0000:1A X     FIRST   64A3          FLAGS   65CB          FOR     0127'    I
FORDEL  0253'         FOREGR  65D0          FORMAT  0000:1B X     FRAGSI  0400
FREE    0000:1C X     FREELS  65E5          FSTDOL  648C          FSTINT  62EC
FWDPTR  65E7          GETDSK  0000:1D X     GETLPT  0000:1E X     GETOPN  0000:1F X
GOSUB   033F'         GOTARG  0408'         GOTO    03F9'    I    GOTO1   040F'
GOTOX   0418'         HCAREA  6593          HELP    0000:20 X     HEXHL   0000:21 X
IF      03B3'    I    INC5IY  0000:22 X     INCRO   65E9          INIDEV  0000:25 X
INITWO  00E5'         INIVDA  0000:23 X     INIVDM  0000:24 X     INIWHL  00E0'
INPTTY  0000:26 X     INPUT   0000:27 X     ISNAME  0000:28 X     ISNL    0000:29 X
JUNK    6542          KBLOCK  65F1          KEYFLG  67FF          KEYPTK  6533
```

CM1 -

+++++ SYMBOL TABLE +++++

KEYTRK	67F7		KISNL	0000:2A	X	KLB	0000:2B	X	LARGE	0000:2C	X
LET	0468'	I	LF	000A		LINE	0000:2D	X	LINEP	0000:2E	X
LISTON	65E2		MACSTU	6536		MACTOP	625E		MAXFRG	0040	
MNMX	65EB		MODIFY	0000:2F	X	NAMADR	0000:30	X	NAMSET	0000:31	X
NBLKB	0000		NBLKM	0001		NEWBOT	6810		NEXT	0269'	I
NL	000A		NLPNT	0000:32	X	NSADDR	6802		NUMBUF	64A3	
NXTABC	0000:33	X	NXTLNK	0000:34	X	NXTPTR	0000:35	X	NXTVAL	0000:36	X
OLDCHR	64A0		OLDCUR	649F		OLDKEY	649D		OLDXY	65EF	
ONEBUF	6729		ONERRO	02C4'		OPRL	0014		OPRSP	655B	
OPRSTK	6547		OPRSZ	655D		OUTCH	0000:37	X	OUTOFF	62E7	
PCNT	655E		PDECBC	0000:38	X	PIXVAL	65ED		POINTE	64A7	
PONOFF	65DA		PRINT	0000:39	X	PRINTF	0000:3A	X	PRINTH	0000:3B	X
PRINTR	6540		PRTYPA	0000:3C	X	PSCAN	0000:3D	X	PUTLPT	0000:3E	X
PUTOPN	0000:3F	X	RAMEND	7FFF		RAMSTR	6900		RANSHT	655F	
REMARK	0454'		RENAME	0000:40	X	RESTOR	0418'	I	RETNON	0000:41	X
RETONE	0000:42	X	RETOUT	0337'		RETURN	030B'	I	RETX	033D'	
RFAKE	033D'		RGBINI	0000:43	X	RMDTMP	6538		RUBACK	0005	
RUBOUT	007F		SAVESP	625C		SCANB	0000:44	X	SCANLA	0000:45	X
SCANNL	0000:46	X	SCLEAR	0000:47	X	SCNEED	0004		SCRWIN	67CD	
SKIP	036F'	I	SMALL	0000:48	X	SNAP	0000:49	X	SOPRSP	653D	
SOPRSZ	653F		STACK	60C8		STAKTO	6000		STOP	0000:4A	X
STREQK	0000:4B	X	STRSIZ	6800		SUBSTU	6534		SWITCH	0000:4C	X
TAB	0009		TABPNT	0000:4D	X	TANGEN	0000:4E	X	TAPBUF	64B3	
TAPCON	64AF		TAPPRO	64B1		TBFEND	6533		TEMPHD	60CA	
TEMPS	62E5		TEXT	0000:4F	X	THEN	0477'	I	THEN1	0486'	
THENX	0492'		TMPARG	67AD		TOP	6818		TTYBEG	6261	
TTYEND	62E1		TTYINT	62E3		TTYPTR	62E1		TXTWIN	67E2	
UARTFL	649C		USEMAP	0000:50	X	USREND	65F1		V3PTR	6573	
VDCHAR	649E		VDNLF	65CF		VIPLH	6545		VOICEO	6563	
WHATSI	0000:51	X	WRMODE	65EE		ZGIM2	0001		ZGREND	681D	
$AADR	0010		$ADDRF	0007		$ADDRI	0005		$ADDRS	0009	
$ANY	001A		$ANYNA	FFFC		$ANYVA	FFFE		$ARGPT	0011	
$BGPTR	000F		$BNDL	0007		$CALLE	000D		$CMDAD	0018	
$CPLBL	002A		$CSBLO	0028		$DATAP	0007		$DOLDE	0001	
$DOLOO	0002		$DVAL	0000		$END	001C		$FADR	000E	
$FLAGS	0002		$FORBL	0024		$FORPT	000B		$FVAL	0006	
$GOSUB	001A		$IADR	000C		$INPBU	0018		$INPPT	0016	
$IVAL	0004		$KEYBL	0026		$LENGT	0001		$LINPT	0005	
$LOCPT	0009		$MIBBL	0022		$MIBEN	001B		$NAMAD	000A	
$NAME	000A		$NASCI	0009		$NDEL	0080		$NLINK	0003	
$NULL	0002		$NVALU	0005		$REPEA	001E		$RVSTU	0013	
$SAME	0020		$SASCI	000A		$SLEN	0006		$STRAD	0008	
$STRPT	0003		$TAF	0000		$TYPE	0000		$USE	0005	
.BLNK.	0000:03	X	.DATA.	0000*	X	.PROG.	0492'	X			

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                       ***
@                     *******
@                         *
@                        *
@                       *
@                      *
@                     *
@                     *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                 THE VERY BEST"
@
@               ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                    (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
          .LINK
          .PREL
          .IDENT  CM2
          .INTERN CORE
          .INTERN DELETE
          .INTERN HELP
          .INTERN INPUT
          .INTERN KBCNLP
          .INTERN PNXT
          .INTERN POUTCH
          .INTERN PRINT
          .INTERN PROMPT
          .INTERN RENAME
          .INTERN WHATSIS


          .EXTERN ABORT
          .EXTERN ACOPY
          .EXTERN ALLOC
          .EXTERN ALSTR
          .EXTERN ARG
          .EXTERN ASSIGN
          .EXTERN CARTYP
          .EXTERN CLEAR
          .EXTERN CLEARIT
          .EXTERN CLEARVAL
          .EXTERN CMDSKO
          .EXTERN CPRINT
          .EXTERN DECSIY
          .EXTERN DELBAG
          .EXTERN DISPLAY
          .EXTERN DOMORE
          .EXTERN ENDSTR
```

```
                        .EXTERN ERRPGM
                        .EXTERN EVAL
                        .EXTERN FINDHASH
                        .EXTERN FINDLAB
                        .EXTERN FREE
                        .EXTERN GETDSK
                        .EXTERN GETLPT
                        .EXTERN GETOPND
                        .EXTERN HEXHL
                        .EXTERN INC5IY
                        .EXTERN INPTTY
                        .EXTERN INT
                        .EXTERN ISNAME
                        .EXTERN ISNL
                        .EXTERN KISNL
                        .EXTERN KLB
                        .EXTERN LN
                        .EXTERN NAMADR
                        .EXTERN NAMSET
                        .EXTERN NLPNT
                        .EXTERN NXTABC
                        .EXTERN NXTLNK
                        .EXTERN NXTPTR
                        .EXTERN NXTVAL
                        .EXTERN OUTCH
                        .EXTERN PDECBC
                        .EXTERN POWER
                        .EXTERN PRINTF
                        .EXTERN PRINTHL
                        .EXTERN PRTYPA
                        .EXTERN PSHOPND
                        .EXTERN PUTOPND
                        .EXTERN RESTORE
                        .EXTERN RETNONE
                        .EXTERN SCANB
                        .EXTERN SCANLAB
                        .EXTERN SCANNL
                        .EXTERN SFREE
                        .EXTERN STREQK
                        .EXTERN SWITCH
                        .EXTERN TABPNT
                        .EXTERN TEMPCHN
                        .EXTERN TEMPUN

                         ; *HELP COMMAND*
                         M.CMD[HELP,..HELX,0,AFIX[H],0,..HEL1][
0000'   18              +HELP:   .BYTE $CMDADR
0001'   0F              +        .BYTE (..HELX-HELP)/16+1
0002'   00              +        .BYTE 0
0003'   635C            +        .WORD AFIX[H][("H"-101Q)*16+FSTINT]
0005'   000F'          +        .WORD ..HEL1
0007'   0000            +        .WORD 0
0009'   48454C5000      +        .ASCIZ /HELP/
                        +]
000E'   00                       .BYTE 0          ;NO SWITCHES
```

```
000F'  CD 0000:23    ..HEL1: CALL KISNL       ;IS EOL
0012'  2052                  JRNZ ..HMES      ;NOPE
0014'  CD 0000:30            CALL PRINTFF
0017'  545950452048          .ASCIZ /TYPE HELP PLUS ONE OF:
002D'  0D0A00        /
0030'  E5                    PUSH H           ;SAVE EOL PTR
0031'  3E41                  MVI A,'A'        ;CHAIN THRU ALL NAMES
0033'  F5            ..DOIT: PUSH PSW         ;SAVE CONTENTS OF A
0034'  CD 0000:17            CALL FINDHASH    ;GET ADDR OF HASH ENTRY
0037'  01 000E               LXI B,14         ;POINT TO COMMANDS
003A'  CD 0000:29            CALL NXTABC      ;GET PTR
003D'  7E            ..CHEK: MOV A,M          ;SEE IF COMMAN
003E'  FE18                  CPI $CMDADR      ;???
0040'  2011                  JRNZ ..MORE      ;END OF COMMANDS
0042'  E5                    PUSH H           ;SAVE PTR TO NAME START
0043'  01 0009               LXI B,$NASCII    ;GET PTR TO ASCII
0046'  09                    DAD B
0047'  CD 0000:31            CALL PRINTHL     ;PRIT WHAT HL-> AT
004A'  CD 0000:3D            CALL TABPNT
004D'  E1                    POP H
004E'  CD 0000:2A            CALL NXTLNK      ;GET NEXT NAME
0051'  18EA                  JMPR ..CHEK
0053'  F1            ..MORE: POP PSW          ;GET HASH CHARACTER BACK
0054'  FE24                  CPI '$'          ;DONE $ YET?
0056'  2809                  JRZ ..EXIT       ;YUP--DONE
0058'  3C                    INR A            ;NO, BUMPSIE
0059'  FE5B                  CPI 'Z'+1        ;DONE WITH Z?
005B'  20D6                  JRNZ ..DOIT      ;NAH
005D'  3E24                  MVI A,'$'        ;DO $ NAMES
005F'  18D2                  JMPR ..DOIT
0061'  CD 0000:28    ..EXIT: CALL NLPNT       ;PRINT NL
0064'  E1                    POP H
0065'  C9                    RET              ;ALL DONE


0066'  FE2C          ..HMES: CPI ','
0068'  2001                  JRNZ ..HM2
006A'  23                    INX H
006B'  CD 0000:27     ..HM2: CALL NAMSET      ;LOOKUP NAME
006E'  AF                    XRA A
006F'  CD 0000:26            CALL NAMADR
                             NZERROR ER.NAME[
0072'  2804           +      JRZ ..0001
0074'  CD 0000:15     +      CALL ERRPGM
0077'  29             +      .BYTE ER.NAME
0078'                 +..0001:]
0078'  E5                    PUSH H  ;SAVE EOL PTR
0079'  EB                    XCHG    ;GET NADDR IN HL
007A'  7E                    MOV A,M
007B'  FE18                  CPI $CMDADR      ;IS COMMAND?
007D'  2052                  JRNZ ..USER      ;NAH
007F'  CD 0000:30            CALL PRINTFF
0082'  434F4D4D414E          .ASCIZ /COMMAND LIKES: /
0092'  01 0007               LXI B,$NASCII-2 ;PRINT NAME
0095'  09                    DAD B            ;POINT AT IT
```

```
0096'   E5                      PUSH H          ;SAVE ARG LIST ADDR
0097'   23                      INX H
0098'   23                      INX H
0099'   CD 0000:31              CALL PRINTHL
009C'   E1                      POP H           ;GET ADDR OF ARG LIST
009D'   CD 0000:2B              CALL NXTPTR     ;IF HL->0, QUIT
00A0'   2012                    JRNZ ..PRIT
00A2'   CD 0000:30              CALL PRINTFF
00A5'   202842592049            .ASCIZ / (BY ITSELF)/
00B2'   18AD            ..EXII: JMPR ..EXIT
00B4'   3E20            ..PRIT: MVI A,' '       ;PRINT SPACE
00B6'   CD 0000:2D              CALL OUTCH
00B9'   7E                      MOV A,M         ;GET ARGLST TYPE
00BA'   B7                      ORA A           ;IF 0, END
00BB'   28A4                    JRZ ..EXIT      ;...
00BD'   FE1E                    CPI $REPEAT
00BF'   28A0                    JRZ ..EXIT
00C1'   FE63                    CPI 99          ;IF 99, SPECIAL
00C3'   2806                    JRZ ..SPEC
00C5'   CD 0000:32              CALL PRTYPA     ;PRINT TYPE IN A
00C8'   23                      INX H
00C9'   18E9                    JMPR ..PRIT     ;CONTINUE
00CB'   23              ..SPEC: INX H
00CC'   CD 0000:31              CALL PRINTHL    ;PRINT SPECIAL STRING
00CF'   1890                    JMPR ..EXIT
00D1'   F5              ..USER: PUSH PSW
00D2'   01 0009                 LXI B,$NASCII
00D5'   09                      DAD B
00D6'   CD 0000:31              CALL PRINTHL
00D9'   CD 0000:30              CALL PRINTFF
00DC'   20495320594F            .ASCIZ / IS YOUR /
00E6'   F1                      POP PSW
00E7'   CD 0000:32              CALL PRTYPA
00EA'   18C6                    JMPR ..EXII
00EC'                   ..HELX:
                        ; *CORE COMMAND*
                        M.CMD[CORE,..CORX,0,CLEAR,0,..COR1][
00EC'   18              +CORE:  .BYTE $CMDADR
00ED'   05              +       .BYTE (..CORX-CORE)/16+1
00EE'   00              +       .BYTE 0
00EF'   0000:0B         +       .WORD CLEAR
00F1'   00FB'           +       .WORD ..COR1
00F3'   0000            +       .WORD 0
00F5'   434F524500      +       .ASCIZ /CORE/
                        +]
00FA'   00                      .BYTE 0         ;NO SWITCHES
00FB'   E5              ..COR1: PUSH    H
00FC'   CD 0000:30              CALL    PRINTFF         ;PRINT HEADER
00FF'   414444522053            .ASCIZ  /ADDR SIZE
0108'   0D0A00          /
010B'   21 65E7                 LXI     H,FWDPTR        ;HL->FIRST FWD PTR
010E'   5E              ..C1:   MOV     E,M             ;DE->NEXT ITEM
010F'   23                      INX     H
0110'   56                      MOV     D,M
0111'   1B                      DCX     D               ;DE->SIZE FIELD
```

```
0112'   1A                      LDAX    D               ;A=BLOCK SIZE
0113'   B7                      ORA     A               ;IS LISTHEAD?
0114'   281E                    JRZ     ..EX
0116'   CD 0000:1D              CALL    HEXHL           ;PRINT OUT ADDR
0119'   2B                      DCX     H               ;1ST HI ORDER
011A'   CD 0000:1D              CALL    HEXHL           ;THEN LO ORDER
011D'   CD 0000:3D              CALL    TABPNT          ;THEN A TAB
0120'   1A                      LDAX    D               ;A=SIZE (BLOCKS)
0121'   2600                    MVI     H,0             ;HL=BLOCK SIZE
0123'   6F                      MOV     L,A
0124'   29                      DAD     H               ;BLOCKS->BYTES
0125'   29                      DAD     H
0126'   29                      DAD     H
0127'   29                      DAD     H
0128'   44                      MOV     B,H             ;BC=BYTE SIZED
0129'   4D                      MOV     C,L
012A'   CD 0000:2E              CALL    PDECBC          ;PRINT THE SIZE
012D'   EB                      XCHG                    ;HL->SIZE
012E'   CD 0000:28              CALL    NLPNT           ;CR,LF
0131'   23                      INX     H               ;HL->FWD PTR AGAIN
0132'   18DA                    JMPR    ..C1            ;DO NEXT ONE
0134'   E1              ..EX:   POP     H
0135'   C9                      RET
0136'                   ..CORX:
                        ; *DELETE COMMAND*
                                M.CMD[DELETE,..DXX,0,DISPLAY,..DELA,..D11][
0136'   18              +DELETE: .BYTE $CMDADR
0137'   03              +        .BYTE (..DXX-DELETE)/16+1
0138'   00              +        .BYTE 0
0139'   0000:12         +        .WORD DISPLAY
013B'   0147'           +        .WORD ..D11
013D'   014A'           +        .WORD ..DELA
013F'   44454C455445+            .ASCIZ /DELETE/
                        +]
0146'   00                      .BYTE 0             ;NO SWITCHES
0147'   CD 0000:08     ..D11:  CALL ARG
014A'   0A1E            ..DELA: .BYTE $NAME,$REPEAT
014C'   E5                      PUSH H
014D'   FD7E00          ..MORE: MOV A,0(Y)          ;GET TYPE
0150'   B7                      ORA A               ;CHECK FER NULL
0151'   280E                    JRZ ..DONE          ;IS ALL DONE
0153'   FD6602                  MOV H,2(Y)          ;GET ADDR
0156'   FD6E01                  MOV L,1(Y)          ;TO DELETE AT
0159'   CD 0000:11              CALL DELBAG         ;DELETE IT
015C'   CD 0000:1E              CALL INC5IY         ;INC IT 3
015F'   18EC                    JMPR ..MORE
0161'   E1              ..DONE: POP H
0162'   C9                      RET
0163'                   ..DXX:
                        ;*INPUT COMMAND FOR INTEGERS*

0163'   CD 0000:23     KBCNLP: CALL KISNL          ;KILL BLANX
0166'   C8                      RZ                  ;RZ ON EOL'S
0167'   FE29                    CPI ')'             ;AND )'S .
0169'   C8                      RZ
```

```
016A'   FE2C                    CPI ',          ;BUT SCAN COMMAS
016C'   C0                      RNZ
016D'   23                      INX H           ;SKIP COMMA
016E'   18F3                    JMPR KBCNLP     ;...

0018                    $INPBUF=$INPPTR+2
                        ;INPUT COMMAND SYNOPSIS
                        ;MIB USAGE
                        ;INPPTR - HOLDS PTR TO DELIM BEFORE CURRENT ARG
                        ;         SO IT KNOWS WHERE TO CONTINUE
                        ;ARGPTR - HOLDS PTR TO CURRENT ARG TO FEED TO
                        ;         INPUT ARGS.  IF $MIBEND->A NULL, THIS
                        ;         IS AN ASCII STRING TO FEED TO
                        ;         EVAL, OTHERWISE IT IS A PASSED ARG LIST
                        ;         IN THE MIB
                        ;INPBUF - HOLDS POINTER TO ARGLIST TYPED IN
                        ;         BY USER VIA INPTTY

                        ;CASE 1--INPUT FROM PASSED ARG LIST IN MIB
                        ;
                        ;EQUATE   VALUE IN    OUT
                        ;                  NOT DONE    DONE
                        ;$INPPTR          0       DELIM   0
                        ;$INPBUF          0       0       0
                        ;$ARGPTR          $MIBEND NEXT    NEXT
                        ;$MIBEND          ARGS    ARGS    ARGS
                        ;CASE 2A--TTY INPUT NEEDED (FIRST TIME CALLED)

                        ;$EQUATE    VALUE  IN      OUT
                        ;                          NOT DONE        DONE
                        ;$INPPTR          0       DELIM           0
                        ;$INPBUF          0       BUFF ADDR       0
                        ;$ARGPTR          0       DELIM           0
                        ;$MIBEND          0       0               0

                         M.CMD[INPUT,..INPX,0,INT,..INPA,..INP1][
0170'   18              +INPUT:  .BYTE $CMDADR
0171'   10              +        .BYTE (..INPX-INPUT)/16+1
0172'   00              +        .BYTE 0
0173'   0000:20         +        .WORD INT
0175'   018D'           +        .WORD ..INP1
0177'   01DC'           +        .WORD ..INPA
0179'   494E50555400+    +        .ASCIZ /INPUT/
                        +]
                         M.SWITCH[NAME,..NAME,..INPA][
017F'   0189'           +        .WORD ..NAME
0181'   01DC'           +        .WORD ..INPA
0183'   4E414D4500      +        .ASCIZ /NAME/
                        +]
0188'   00                       .BYTE 0           ;END OF SWITCHES
0189'   3E0A             ..NAME: MVI A,$NAME       ;ASK FOR NAME
018B'   1802                     JMPR ..INPN
018D'   3E06             ..INP1: MVI A,$FVAL
018F'   DD2A 625E        ..INPN: LIXD MACTOP       ;LOAD IX
0193'   DD7714                   MOV $RVSTUF+1(X),A      ;SAVE TYPE
```

```
0196'  EB                    XCHG              ;SAVE HL IN DE
0197'  DD6617                MOV H,$INPPTR+1(X)
019A'  DD6E16                MOV L,$INPPTR(X);GET INPUT LINE PTR
019D'  7C                    MOV A,H           ;IS ZERO?
019E'  B5                    ORA L             ;IF SO, NEW INPUT COMMAND
019F'  2001                  JRNZ ..INPP       ;ELSE USE IT AS PTR
01A1'  EB                    XCHG              ;GET HL BACK
01A2'  E5            ..INPP: PUSH H            ;SAVE PTR INTO LINE
01A3'  CD 0163'              CALL KBCNLP       ;SCAN BLANX, COMMANS
01A6'  2031                  JRNZ ..GET        ;CLOSE ) AND EOL'S
01A8'  E5            ..DONE: PUSH H            ;DONE!, CLEANUP $INPBUF
01A9'  DD6E18                MOV L,$INPBUF(X)          ;IF ALLOC'D
01AC'  DD6619                MOV H,$INPBUF+1(X)
01AF'  7C                    MOV A,H           ;IS THERE?
01B0'  B5                    ORA L
01B1'  280A                  JRZ ..DXX         ;NAH
01B3'  01 FFFB               LXI B,-5          ;ADDR IS ALLOC'D BLOCK
01B6'  09                    DAD B             ;PLUS 5
01B7'  CD 0000:3F            CALL TEMPUN       ;TEM UNLINK IT
01BA'  CD 0000:19            CALL FREE         ;RESTORE IT
01BD'  E1            ..DXX:  POP H             ;EOL POINTER
01BE'  AF                    XRA A             ;CLEAR A
01BF'  DD7716                MOV $INPPTR(X),A
01C2'  DD7717                MOV $INPPTR+1(X),A
01C5'  DD7718                MOV $INPBUF(X),A
01C8'  DD7719                MOV $INPBUF+1(X),A
01CB'  DD7E1B                MOV A,$MIBEND(X)          ;ZAP ONLY IF
01CE'  B7                    ORA A             ;MIBEND=$TAF
01CF'  2006                  JRNZ ..OUT
01D1'  DD7711                MOV $ARGPTR(X),A
01D4'  DD7712                MOV $ARGPTR+1(X),A        ;CLEAR ICKIES
01D7'  F1            ..OUT:  POP PSW           ;TRASH ENTRY POINTER
01D8'  C9                    RET               ;AND GO BYEBYE
01D9'  CD 0000:08    ..GET:  CALL ARG
01DC'  0A00          ..INPA: .BYTE $NAME,$TAF
01DE'  CD 0000:1E            CALL INC5IY       ;FOR ASSIGN BELOW
01E1'  E5                    PUSH H
01E2'  DD6E11                MOV L,$ARGPTR(X)          ;GET ARG LIST
01E5'  DD6612                MOV H,$ARGPTR+1(X)        ;SET BY DOMACRO
01E8'  7C                    MOV A,H           ;SEE IF ZILCH
01E9'  B5                    ORA L
01EA'  284B                  JRZ ..TTY         ;IT IS, NEED HUMAN NOW
01EC'  7E                    MOV A,M           ;IS $MIBEND =0?
01ED'  B7                    ORA A             ;OR IS ARGS DONE FOR?
01EE'  2847                  JRZ ..TTY         ;YUP-DO HUMAN INPUT
01F0'  DD7E1B                MOV A,$MIBEND(X)          ;CHECK THIS TOO
01F3'  B7                    ORA A             ;IF ZILCH, ARGS IS ASCII
01F4'  282C                  JRZ ..TTA         ;FROM MULTI-HUMAN INPUT
01F6'  E5                    PUSH H            ;FER SURE GOT PASSED ARG
01F7'  FDE3                  XTIY              ;LIST, POINT IY AT IT
01F9'  CD 0000:1C            CALL GETOPND      ;GET STUFF INTO A&DE FOR
01FC'  CD 0000:1E            CALL INC5IY       ;ASSIGN BELOW
01FF'  FDE3                  XTIY              ;GET EVAL'S IY BACK
0201'  E1                    POP H             ;AND PTR INTO HL
0202'  CD 0000:34    ..STUF: CALL PUTOPND      ;STUFF DE&A INTO IY
```

```
0205'   7E                      MOV A,M         ;IS ->NULL?
0206'   B7                      ORA A
0207'   2002                    JRNZ ..OK       ;NO
0209'   67                      MOV H,A         ;ZERO IT
020A'   6F                      MOV L,A         ;ARGPTR, THAT IS
020B'   DD7511          ..OK:   MOV $ARGPTR(X),L        ;POINT AT NEXT
020E'   DD7412                  MOV $ARGPTR+1(X),H      ;ARG IN LIST
0211'   CD 0000:09             CALL ASSIGN     ;DO ASSIGNMENT
0214'   E1                      POP H           ;GET TERMY ADDR BACK
0215'   DD7417                  MOV $INPPTR+1(X),H      ;STORE IT
0218'   DD7516                  MOV $INPPTR(X),L
021B'   CD 0000:10             CALL DEC5IY     ;RESET IY
021E'   F1                      POP PSW         ;TRASH OLD TERMY ADDR
021F'   C3 01A2'                JMP ..INPP

0222'   CD 0163'       ..TTA:  CALL KBCNLP     ;SCAN TO NEXT ARG IN LIST
0225'   2810                    JRZ ..TTY       ;NONE-GET INPUT

0227'   DD7E14          ..TARG: MOV A,$RVSTUF+1(X)      ;GET TYPE WANTED
022A'   CD 0000:16             CALL EVAL       ;INTO A&DE
022D'   CD 0000:1C             CALL GETOPND    ;GET IT
0230'   FE0A                  CPI $NAME       ;IS INP.NAME?
0232'   CC 0264'               CZ CNVNAM       ;DO STRING MASSAGE
0235'   18CB                    JMPR ..STUF     ;AND SETUP ARGPTR, ETC.

0237'   DD6619          ..TTY:  MOV H,$INPBUF+1(X)      ;GET BUF ADR
023A'   DD6E18                  MOV L,$INPBUF(X)
023D'   7C                      MOV A,H
023E'   B5                      ORA L
023F'   2018                    JRNZ ..ISBF     ;GOT ONE ALREADY
0241'   3E08                    MVI A,8         ;GET 9 BLOX
0243'   CD 0000:06             CALL ALLOC
0246'   CD 0000:3E             CALL TEMPCHN    ;ALLOC&TEMPCHAIN
0249'   01 0005                LXI B,5         ;ADD 5 TO POINT
024C'   09                      DAD B           ;AT BUFFER COUNT
024D'   01 0080                 LXI B,128       ;CLEAR STUFF
0250'   CD 0000:0C             CALL CLEARIT
0253'   DD7419                  MOV $INPBUF+1(X),H      ;SAVE ADDR
0256'   DD7518                  MOV $INPBUF(X),L
0259'   CD 0000:1F    ..ISBF:   CALL INPTTY     ;GET HUMAN TO
025C'   23                      INX H           ;TYPE SOMETHING
025D'   28C8                    JRZ ..TARG      ;JUMP OVER COUNT
025F'   F1                      POP PSW         ;NO NL YET
0260'   F1                      POP PSW         ;FAIL
0261'   C3 0000:0E              JMP CMDSKO
0264'                   ..INPX:
                        ;++++
                        ;
                        ; CNVNAM
                        ; CONVERTS A NAME BLOCK TO A STRING
                        ;
                        ; NEEDS:
                        ;   DE -> TOP OF NAME BLOCK
                        ;       $NASCII - ASCII NAME FIELD
                        ;
```

```
                        ; RETURNS:
                        ;   A = TYPE STRING ($STRADR)
                        ;   DE -> TOP OF STRING WHICH CONTAINS THE ASCII NAME
                        ;
                        ; CALLS:
                        ;   ALSTR - ALLOCATE STRING
                        ;   ENDSTR - CLEAN UP AND RETURN
                        ;
                        ;----
0264'  E5               CNVNAM: PUSH    H               ;SAVE REGS
0265'  C5                       PUSH    B
0266'  21 0009                  LXI     H,$NASCII       ;OFFSET OF NAME
0269'  19                       DAD     D               ;HL->NAME FIELD
026A'  E5                       PUSH    H               ;SAVE PTR TO NAME
026B'  CD 0000:07               CALL    ALSTR           ;ALLOCATE STRING
026E'  E3                       XTHL                    ;(SP)->STRING,HL->NAME
026F'  EDA0             ..C1:   LDI                     ;COPY A CHAR
0271'  E2 027F'                 JPO ..ERR               ;OUT O'CORE
0274'  7E                       MOV     A,M             ;GET NEXT CHAR
0275'  B7                       ORA     A               ;IF NULL, END OF NAME
0276'  20F7                     JRNZ    ..C1            ;IF NOT, DO ANOTHER
0278'  E1                       POP     H               ;HL->TOP OF STRING
0279'  CD 0000:14               CALL    ENDSTR          ;GARBAGE COLLECT LEFTOVE
                       R
027C'  C1                       POP     B               ;RESTORE REGS
027D'  E1                       POP     H
027E'  C9                       RET
027F'                   ..ERR:  ERROR ER.COR[
027F'  CD 0000:15       +       CALL ERRPGM
0282'  1B               +       .BYTE ER.COR
                        +]

                        ; *WHATSIS COMMAND--GETS TYPE IN VAR PASSED

                        M.CMD[WHATSIS,..WHX,0,AFIX[W],..WHA,..WHA1][
0283'  18               +WHATSIS:       .BYTE $CMDADR
0284'  04               +       .BYTE (..WHX-WHATSIS)/16+1
0285'  00               +       .BYTE 0
0286'  644C             +       .WORD AFIX[W][("W"-101Q)*16+FSTINT]
0288'  0295'            +       .WORD ..WHA1
028A'  0298'            +       .WORD ..WHA
028C'  574841545349+    +       .ASCIZ /WHATSIS/
                        +]
0294'  00                       .BYTE 0         ;NO SWITCHES
0295'  CD 0000:08       ..WHA1: CALL ARG
0298'  0A1E             ..WHA:  .BYTE $NAME,$REPEAT
029A'  E5                       PUSH H
029B'  FD6E01                   MOV L,1(Y)
029E'  FD6602                   MOV H,2(Y)
02A1'  7E                       MOV A,M         ;GET TYPE
02A2'  FE0A                     CPI $NAMADR     ;IS NAME?
02A4'  CC 0000:2C               CZ NXTVAL       ;IF SO, GET VALUE
02A7'  5E                       MOV E,M         ;GET TYPE
02A8'  1600                     MVI D,0         ;IN DE
02AA'  FD7E03                   MOV A,3(Y)      ;SEE IF NAME PASSED
```

```
02AD'   B7                      ORA A
02AE'   3E04                    MVI A,$IVAL     ;USEFUL TAG
02B0'   2004                    JRNZ ..CMD      ;YUP, USED AS CMD
02B2'   E1                      POP H           ;GET TERMY BACK
02B3'   C3 0000:36              JMP RETNONE     ;RETURN VALUE
02B6'   CD 0000:1E      ..CMD:  CALL INC5IY
02B9'   CD 0000:33              CALL PSHOPND    ;STICK IT
02BC'   CD 0000:09              CALL ASSIGN     ;STICK IT
02BF'   E1                      POP H
02C0'   C9                      RET
02C1'                   ..WHX:

                        ;++++
                        ;
                        ; PRSTR
                        ; PRINTS AN ASCII STRING ON THE TERMINAL
                        ;
                        ; NEEDS:
                        ;   DE -> TOP OF STRING HEADER
                        ;       STRING DATA STARTS AS $SASCII+DE,
                        ;       IS NULL TERMINATED
                        ;       DESTROYS DE
                        ;
                        ; CALLS:
                        ;   OUTCH - OUTPUT A CHAR
                        ;   SFREE - ATTEMPTS TO FREE STRING
                        ;
                        ;----
02C1'                   PRSTR:
02C1'   7A                      MOV     A,D             ;IF NULL, JUST
02C2'   83                      ADD     E               ;RETURN
02C3'   C8                      RZ
02C4'   E5                      PUSH    H               ;SAVE REGS
02C5'   21 000A                 LXI     H,$SASCII       ;GET STRING DATA
02C8'   19                      DAD     D               ;HL->1ST CHAR
02C9'   3A 62E7                 LDA OUTOFF              ;SUPPRESS FROM
02CC'   B7                      ORA A                   ;INPUT ARGS
02CD'   200A                    JRNZ ..EX               ;BEING THERE?
02CF'   7E              ..P1:   MOV     A,M             ;END OF STRING?
02D0'   B7                      ORA     A
02D1'   2806                    JRZ     ..EX            ;EXIT THEN
02D3'   CD 0366'                CALL    POUTCH          ;OUTPUT A CHAR
02D6'   23                      INX     H               ;HL->NEXT CHAR
02D7'   18F6                    JMPR    ..P1            ;LOOP AGAIN
02D9'   EB              ..EX:   XCHG                    ;HL->TOP OF STRING
02DA'   CD 0000:3A              CALL    SFREE           ;FREE IT
02DD'   E1                      POP     H               ;RESTORE PTR
02DE'   C9                      RET
                        ;++++
                        ;
                        ; PRINT
                        ; COMMAND TO PRINT STRINGS ON THE TERMINAL
                        ; A SPACE IS PRINTED AFTER EACH ARGUMENT AND A NEWLINE
                        ; IS PRINTED AT THE END OF THE LINE. UNLIKE PROMPT,
                        ; PRINT WILL DISPLAY ITS ARGUMENTS REGARDLESS OF WHETHEF
```

```
                        ; THE MACRO ARGUMENT LIST IS EXHAUSTED OR NOT
                        ;
                        ; SYNTAX:
                        ;       PRINT <STRING>,<STRING>,...
                        ;
                        ;----
                                M.CMD[PRINT,..PEND,0,PROMPT,..PARG,..PRNT][
02DF'   18              +PRINT: .BYTE $CMDADR
02E0'   02              +       .BYTE (..PEND-PRINT)/16+1
02E1'   00              +       .BYTE 0
02E2'   02FA'           +       .WORD PROMPT
02E4'   02EF'           +       .WORD ..PRNT
02E6'   02F2'           +       .WORD ..PARG
02E8'   5052494E5400+   +       .ASCIZ /PRINT/
                        +]
02EE'   00                      .BYTE   0
02EF'   CD 0000:08      ..PRNT: CALL    ARG
02F2'   081E            ..PARG: .BYTE   $STRADR,$REPEAT
02F4'   CD 0322'                CALL    PNXT            ;PRINT ARGUMENTS
02F7'   C3 0000:28              JMP     NLPNT           ;AND NEWLINE
02FA'                   ..PEND:;++++
                        ;
                        ; PROMPT
                        ; COMMAND TO PROMPT THE USER WITH A MESSAGE (OR MESSAGES
                        ) ON
                        ; THE TERMINAL. UNLIKE PRINT, PROMPT WILL BE SUPPRESSED
                        IF THERE
                        ; ARE ANY ARGUMENTS LEFT IN THE MACRO'S ARGUMENT LIST WH
                        ICH HAVE
                        ; NOT BEEN PICKED UP BY INPUT. A TRAILING NEWLINE IS NOT
                         ADDED,
                        ; BUT SPACES ARE PUT BETWEEN THE STRING ARGUMENTS
                        ;
                        ; SYNTAX:
                        ;       PROMPT <STRING>,<STRING>,...
                        ;
                        ;----
                                M.CMD[PROMPT,..PEND,0,PRLPT,..PARG,..PROM][
02FA'   18              +PROMPT: .BYTE $CMDADR
02FB'   03              +       .BYTE (..PEND-PROMPT)/16+1
02FC'   00              +       .BYTE 0
02FD'   033B'           +       .WORD PRLPT
02FF'   030B'           +       .WORD ..PROM
0301'   030E'           +       .WORD ..PARG
0303'   50524F4D5054+   +       .ASCIZ /PROMPT/
                        +]
030A'   00                      .BYTE   0
030B'   CD 0000:08      ..PROM: CALL    ARG
030E'   081E            ..PARG: .BYTE   $STRADR,$REPEAT
0310'   DD2A 625E              LIXD    MACTOP          ;IX->MACRO HEADER
0314'   DD7E00                  MOV     A,0(X)          ;CHECK IF $MIBBLK
0317'   FE22                  CPI     $MIBBLK         ;IS MIB?
0319'   2007                  JRNZ    PNXT            ;NO? PRINT ANYWAY
031B'   DDA61B                ANA     $MIBEND(X)      ;ANY ARGS?
```

```
031E'  CC 0322'            CZ      PNXT            ;NO? PRINT
0321'  C9                  RET
0322'                  ..PEND:
                       ;++++
                       ;
                       ; PNXT
                       ; PRINTS THE ARGUMENTS LIST OF STRINGS ON THE IY STACK
                       ; FOR THE PRINT AND PROMPT COMMANDS
                       ;
                       ; NEEDS:
                       ;   IY -> LIST OF STRING ARGUMENTS
                       ;
                       ; RETURNS:
                       ;   STRINGS PRINTED OUT ON THE TERMINAL WITH SPACES BETW
                       EEN
                       ;   THEM, IY, DE DESTROYED
                       ;
                       ; CALLS:
                       ;   OUTCH - OUTPUT A CHAR
                       ;   PRSTR - PRINT A STRING
                       ;   INC5IY - GET NEXT ARG
                       ;
                       ;----
0322'  FD7E00          PNXT:   MOV     A,E.TYP(Y)      ;CHECK TYPE
0325'  FE00                    CPI     $TAF            ;NO MORE ARGS?
0327'  C8                      RZ
0328'  FD5602                  MOV     D,E.HVAL(Y)     ;DE -> STRING TO PRINT
032B'  FD5E01                  MOV     E,E.LVAL(Y)
032E'  CD 0000:1E              CALL    INC5IY          ;GET NEXT ARG
0331'  CD 02C1'                CALL    PRSTR           ;PRINT STRING
0334'  3E20                    MVI     A,' '           ;PRINT BLANK
0336'  CD 0366'                CALL    POUTCH
0339'  18E7                    JMPR    PNXT            ;CONTINUE
                       ; ** CHANGE **
                       M.CMD[PRLPT,..PEND,0,POWER,..PARG,..PST][
033B'  18              +PRLPT:  .BYTE  $CMDADR
033C'  03              +        .BYTE  (..PEND-PRLPT)/16+1
033D'  00              +        .BYTE  0
033E'  0000:2F         +        .WORD  POWER
0340'  034A'           +        .WORD  ..PST
0342'  034D'           +        .WORD  ..PARG
0344'  50524C505400    +        .ASCIZ /PRLPT/
                       +]
034A'  CD 0000:08      ..PST:   CALL    ARG
034D'  081E            ..PARG:  .BYTE   $STRADR,$REPEAT
034F'  E5                       PUSH    H
0350'  21 0000:0F               LXI     H,CPRINT
0353'  22 6540                  SHLD    PRINTR
0356'  CD 0322'                 CALL    PNXT
0359'  3E0A                     MVI     A,NL
035B'  CD 0366'                 CALL    POUTCH
035E'  21 0000:2D               LXI     H,OUTCH
0361'  22 6540                  SHLD    PRINTR
0364'  E1                       POP     H
0365'  C9                       RET
```

```
0366'                   ..PEND:
0366'   E5              POUTCH: PUSH    H
0367'   2A 6540                 LHLD    PRINTR
036A'   E3                      XTHL
036B'   C9                      RET             ; CALL VIA RETURN
                        ; ** DONE WITH CHANGES **
                        ;+
                        ;  *RENAME COMMAND*
                        ;+
                        ;++++
                        ;
                        ; RENAME <OLD>,<NEW>
                        ; RENAMES THE OLD VALUE WITH THE NEW NAME
                        ; <OLD> AND <NEW> ARE BOTH ZGRASS NAMES
                        ; <OLD> MUST BE A NAME WHICH IS ALREADY THERE,
                        ; <NEW> MUST BE A NAME WHICH DOES NOT YET EXIST
                        ; IF <OLD> IS A NON-DELETEABLE NAME, AN ERROR IS
                        ; PRINTED. ONLY THE TYPE AND VALUE FIELDS ARE
                        ; COPIED
                        ;
                        ; NEEDS:
                        ;   ON ARGUMENT STACK:
                        ;       TYPE ($NAME)
                        ;       ADDR OF OLD NAME
                        ;       TYPE ($NAME)
                        ;       ADDR OF NEW NAME
                        ;
                        ; CALLS:
                        ;   CLEARVAL - CLEAR VALUE & TYPE
                        ;
                        ;----
                         M.CMD[RENAME,..RENX,0,RESTORE,..RENA,..REN1][
036C'   18              +RENAME: .BYTE $CMDADR
036D'   06              +        .BYTE (..RENX-RENAME)/16+1
036E'   00              +        .BYTE 0
036F'   0000:35         +        .WORD RESTORE
0371'   038B'           +        .WORD ..REN1
0373'   037D'           +        .WORD ..RENA
0375'   52454E414D45+    +        .ASCIZ /RENAME/
                        +]
037C'   00                      .BYTE 0         ;NO SWITCHES
037D'   63              ..RENA: .BYTE 99        ;SPECIAL
037E'   4E414D452C4E            .ASCIZ /NAME,NEWNAME/
038B'   CD 0000:08      ..REN1: CALL    ARG             ;GET ARGS
038E'   0A0A00                  .BYTE   $NAME,$NAME,$TAF
0391'   E5                      PUSH H          ;SAVE TERMY
0392'   FD6E01                  MOV     L,1(Y)          ;HL->OLD NAME
0395'   FD6602                  MOV     H,2(Y)
0398'   E5                      PUSH H
0399'   DDE1                    POP X                   ;IX-> OLD NAME
039B'   DDCB027E                BIT     $BNDL,$FLAGS(X) ;NO-DELETE?
                                NZERROR ER.REN          ;NO? RENAME ERROR[
039F'   2804            +       JRZ ..0002
03A1'   CD 0000:15       +       CALL ERRPGM
03A4'   2C               +       .BYTE ER.REN
```

```
03A5'                  +..0002:]
03A5'   FD5E06                 MOV     E,6(Y)          ;DE->NEW NAME
03A8'   FD5607                 MOV     D,7(Y)
03AB'   1A                     LDAX    D               ;A=TYPE OF NEW
03AC'   FE02                   CPI     $NULL           ;ALREADY THERE?
                               NZERROR ER.NAE          ;NAME ALREADY EXISTS[
03AE'   2804           +       JRZ     ..0003
03B0'   CD 0000:15     +       CALL    ERRPGM
03B3'   2D             +       .BYTE   ER.NAE
03B4'                  +..0003:]
03B4'   7E                     MOV     A,M             ;A=TYPE OF OLD
03B5'   12                     STAX    D               ;SAVE IT
03B6'   E5                     PUSH    H               ;SAVE NAME PTR
03B7'   21 0005                LXI     H,$NVALUE       ;HL=OFFSET OF VAL
03BA'   19                     DAD     D               ;HL->NEW VALUE
03BB'   DD7E05                 MOV     A,$NVALUE(X)    ;MOVE IN NEW VAL
03BE'   77                     MOV     M,A             ;THE HARD WAY
03BF'   23                     INX     H
03C0'   DD7E06                 MOV     A,$NVALUE+1(X)
03C3'   77                     MOV     M,A
03C4'   E1                     POP     H               ;HL->TOP
03C5'   CD 0000:0D             CALL    CLEARVAL                ;CLEAR OLD VAL
03C8'   E1                     POP     H
03C9'   C9                     RET
03CA'                   ..RENX:
                        .END
```

CM2 -
+++++ SYMBOL TABLE +++++

AASN	005F	ABORT	0000:04 X	ACOPY	0000:05 X	ALLOC	0000:06 X
ALSTR	0000:07 X	ARG	0000:08 X	ARGSTK	60CC	ASSIGN	0000:09 X
BACKGR	65CC	BLANK	0020	BOTRAM	6000	BOTTOM	64A9
CARTYP	0000:0A X	CHARSL	65DD	CLEAR	0000:0B X	CLEARI	0000:0C X
CLEARS	60CA	CLEARV	0000:0D X	CMDSKO	0000:0E X	CNTRL	000C
CNTRLC	65CD	CNTRLO	65D9	CNTRLU	0015	CNTRLZ	65E4
CNVNAM	0264'	CORE	00EC' I	CPLARE	611C	CPLSIZ	0140
CPRINT	0000:0F X	CR	000D	CSBLOK	668D	CSFLAG	6260
CURCX	6814	CURCY	6816	CURREN	64A5	C.CO	0011
C.C1	0012	C.CX	000B	C.CY	000D	C.DP	0013
C.ST	0002	C.X	0003	C.XF	000F	C.XS	0007
C.Y	0005	C.YF	0010	C.YS	0009	DDTON	65CE
DEC5IY	0000:10 X	DELBAG	0000:11 X	DELETE	0136' I	DEVBL	6589
DEVCL0	6579	DEVCL1	657B	DEVCL2	657D	DEVCL3	657F
DEVCL4	6581	DEVCL5	6583	DEVCL6	6585	DEVCL7	6587
DEVFB	65CB	DEVHCB	658F	DEVMO	658B	DEVNM	65B7
DEVNT	658D	DEVTNA	65BB	DEVTNB	65BF	DEVTNC	65C3
DEVVA	65BD	DEVVAR	6579	DEVVB	65C1	DEVVBL	6591
DEVVC	65C5	DEVVD	65C9	DEVVN	65B9	DEVVS	65C7
DEVXCD	65B3	DEVYCD	65B5	DISPLA	0000:12 X	DOLPLH	62E8
DOLPPT	62EA	DOMORE	0000:13 X	DUMBST	6577	EDBCNT	64AD
EDCCNT	64A7	EDLONG	6812	EDMODE	681C	EDNAME	64A1
EDNCX	680C	EDNCY	680E	EDNEWS	64A5	EDOCX	6808
EDOCY	680A	EDPN	6806	EDPO	6804	EDPTRC	64AB
EDPTRL	64A9	EDSTR	681A	ENDSTR	0000:14 X	ERABIT	0002
ERRPGM	0000:15 X	ER.ARA	002F	ER.ARG	0034	ER.ASN	0015
ER.BOX	001A	ER.CHN	0002	ER.CMD	001F	ER.CNV	0016
ER.COR	001B	ER.CTL	0036	ER.DEL	0026	ER.DIM	0030
ER.DIV	0018	ER.DP	0037	ER.DSK	0019	ER.EDT	0035
ER.FMT	0038	ER.FNF	001C	ER.FOR	0028	ER.IMP	0003
ER.LAB	0025	ER.MAC	0022	ER.NAE	002D	ER.NAM	0029
ER.NEG	003A	ER.NOT	0023	ER.NUL	0039	ER.NUM	002B
ER.NXT	001D	ER.OFL	0017	ER.OPN	0014	ER.OVE	001E
ER.PAR	002A	ER.REN	002C	ER.RET	0024	ER.SEP	0021
ER.SNP	0031	ER.SPC	002E	ER.STK	0004	ER.SW	0032
ER.TER	0020	ER.UFL	0033	ER.UNF	0027	EVAL	0000:16 X
EXTDEL	002E	E.HVAL	0002	E.LVAL	0001	E.SIZ	0005
E.TYP	0000	E.VAL	0001	FCNTH	65D2	FCNTI	65D3
FCNTJ	65D4	FCNTK	65D5	FCNTL	65D6	FCNTV	65E0
FCNTY	65E3	FINDHA	0000:17 X	FINDLA	0000:18 X	FIRST	64A3
FLAGS	65CB	FOREGR	65D0	FRAGSI	0400	FREE	0000:19 X
FREELS	65E5	FSTDOL	648C	FSTINT	62EC	FWDPTR	65E7
GETDSK	0000:1A X	GETLPT	0000:1B X	GETOPN	0000:1C X	HCAREA	6593
HELP	0000' I	HEXHL	0000:1D X	INC5IY	0000:1E X	INCRO	65E9
INPTTY	0000:1F X	INPUT	0170' I	INT	0000:20 X	ISNAME	0000:21 X
ISNL	0000:22 X	JUNK	6542	KBCNLP	0163' I	KBLOCK	65F1
KEYFLG	67FF	KEYPTK	6533	KEYTRK	67F7	KISNL	0000:23 X
KLB	0000:24 X	LF	000A	LISTON	65E2	LN	0000:25 X
MACSTU	6536	MACTOP	625E	MAXFRG	0040	MNMX	65EB
NAMADR	0000:26 X	NAMSET	0000:27 X	NBLKB	0000	NBLKM	0001
NEWBOT	6810	NL	000A	NLPNT	0000:28 X	NSADDR	6802
NUMBUF	64A3	NXTABC	0000:29 X	NXTLNK	0000:2A X	NXTPTR	0000:2B X
NXTVAL	0000:2C X	OLDCHR	64A0	OLDCUR	649F	OLDKEY	649D
OLDXY	65EF	ONEBUF	6729	OPRL	0014	OPRSP	655B
OPRSTK	6547	OPRSZ	655D	OUTCH	0000:2D X	OUTOFF	62E7

CM2 -
+++++ SYMBOL TABLE +++++

```
PCNT    655E          PDECBC  0000:2E X     PIXVAL  65ED          PNXT    0322'    I
POINTE  64A7          PONOFF  65DA          POUTCH  0366'    I     POWER   0000:2F X
PRINT   02DF'    I    PRINTF  0000:30 X     PRINTH  0000:31 X     PRINTR  6540
PRLPT   033B'         PROMPT  02FA'    I     PRSTR   02C1'         PRTYPA  0000:32 X
PSHOPN  0000:33 X     PUTOPN  0000:34 X     RAMEND  7FFF          RAMSTR  6900
RANSHT  655F          RENAME  036C'    I     RESTOR  0000:35 X     RETNON  0000:36 X
RMDTMP  6538          RUBACK  0005          RUBOUT  007F          SAVESP  625C
SCANB   0000:37 X     SCANLA  0000:38 X     SCANNL  0000:39 X     SCNEED  0004
SCRWIN  67CD          SFREE   0000:3A X     SOPRSP  653D          SOPRSZ  653F
STACK   60C8          STAKTO  6000          STREQK  0000:3B X     STRSIZ  6800
SUBSTU  6534          SWITCH  0000:3C X     TAB     0009          TABPNT  0000:3D X
TAPBUF  64B3          TAPCON  64AF          TAPPRO  64B1          TBFEND  6533
TEMPCH  0000:3E X     TEMPHD  60CA          TEMPS   62E5          TEMPUN  0000:3F X
TMPARG  67AD          TOP     6818          TTYBEG  6261          TTYEND  62E1
TTYINT  62E3          TTYPTR  62E1          TXTWIN  67E2          UARTFL  649C
USREND  65F1          V3PTR   6573          VDCHAR  649E          VDNLF   65CF
VIPLH   6545          VOICEO  6563          WHATSI  0283'    I     WRMODE  65EE
ZGIM2   0001          ZGREND  681D          $AADR   0010          $ADDRF  0007
$ADDRI  0005          $ADDRS  0009          $ANY    001A          $ANYNA  FFFC
$ANYVA  FFFE          $ARGPT  0011          $BGPTR  000F          $BNDL   0007
$CALLE  000D          $CMDAD  0018          $CPLBL  002A          $CSBLO  0028
$DATAP  0007          $DOLDE  0001          $DOLOO  0002          $DVAL   0000
$END    001C          $FADR   000E          $FLAGS  0002          $FORBL  0024
$FORPT  000B          $FVAL   0006          $GOSUB  001A          $IADR   000C
$INPBU  0018          $INPPT  0016          $IVAL   0004          $KEYBL  0026
$LENGT  0001          $LINPT  0005          $LOCPT  0009          $MIBBL  0022
$MIBEN  001B          $NAMAD  000A          $NAME   000A          $NASCI  0009
$NDEL   0080          $NLINK  0003          $NULL   0002          $NVALU  0005
$REPEA  001E          $RVSTU  0013          $SAME   0020          $SASCI  000A
$SLEN   0006          $STRAD  0008          $STRPT  0003          $TAF    0000
$TYPE   0000          $USE    0005          .BLNK.  0000:03 X     .DATA.  0000*    X
.PROG.  03CA'    X
```

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                       ***
@                     *******
@                         *
@                        *
@                       *
@                      *
@                     *
@                     *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                THE VERY BEST"
@
@              ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                    (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
          .LINK
          .PREL
          .IDENT  NAM
          .EXTERN AFREE
          .EXTERN ALLOC
          .EXTERN BALLY
          .EXTERN CARTYA
          .EXTERN CARTYP
          .EXTERN CHKAZ
          .EXTERN CLEARIT
          .EXTERN ERRPGM
          .EXTERN EVAL
          .EXTERN FREE
          .EXTERN GETLPT
          .EXTERN INC5IY
          .EXTERN INPTTY
          .EXTERN KISNL
          .EXTERN NLPNT
          .EXTERN NXTABC
          .EXTERN NXTLNK
          .EXTERN NXTVAL
          .EXTERN OUTCH
          .EXTERN PRINTA
          .EXTERN PRINTFF
          .EXTERN PRINTHL
          .EXTERN PUTOPND
          .EXTERN SFREE
          .XTERN TABPNT
          .EXTERN TEMPCHN
          .EXTERN TEMPUNCH
          .INTERN ARG
          .INTERN CLEARVAL
          .INTERN DELBAG
```

```
                                .INTERN FINDHASH
                                .INTERN ISNAME
                                .INTERN NAMADR
                                .INTERN NAMBLD
                                .INTERN NAMSET
                                .INTERN PRTYPA
                                .INTERN PSCAN

                        ;FINDHASH--GETS ADDR OF A-Z VAR INTO HL

                        ;IN     A HAS CHARACTER IN IT
                        ;OUT    HL->FWA OF HASH VARIABLE (A-Z)

0000'  C5               FINDHASH:       PUSH B
0001'  2600                     MVI H,0
0003'  FE24                     CPI '$'         ;IS $?
0005'  2005                     JRNZ ..NDOL     ;NO--ALPHA
0007'  21 648C                  LXI H,FSTDOL    ;FWA OF THIS GUY
000A'  180B                     JMPR ..FINO     ;AND EXIT
000C'  D641             ..NDOL: SUI 'A'         ;GET OFFSET FROM
                                                ;CHAR IN A
000E'  6F                       MOV L,A         ;PUT RMNDR IN L
000F'  29                       DAD H           ;MUL BY 16
0010'  29                       DAD H
0011'  29                       DAD H
0012'  29                       DAD H
0013'  01 62EC                  LXI B,FSTINT    ;GET ADDR OF FIRST ONE
0016'  09                       DAD B           ;AND ADD IT IN
0017'  C1               ..FINO: POP B
0018'  C9                       RET

                        ;++++
                        ; *DELBAG--DELETES STUFF BY TYPE*
                        ;
                        ;IN:    FWA OF NAMEBLOCK IN HL
                        ;OUT:   HL SAME, THING & VALUE DELETED
                        ;NO ACTION IF HL=0 OR HL-> 0 TYPE
                        ;
0019'  7C               DELBAG: MOV A,H
001A'  B5                       ORA L
001B'  C8                       RZ              ;IGNORE IF HL=0
001C'  E5                       PUSH H
001D'  C5                       PUSH B
001E'  E5                       PUSH H          ;PASS TO ROUTINE VIA XTHL
001F'  4E                       MOV C,M         ;TYPE
0020'  0600                     MVI B,0         ;COMPUTE DISPATCH
0022'  21 002A'                 LXI H,..DTAB    ;ADDRESS
0025'  CD 0000:13               CALL NXTABC     ;LIKE THIS
0028'  E3                       XTHL            ;SWITCH 'EM
0029'  C9                       RET             ;AND GOTO IT
002A'  0063'            ..DTAB: .WORD ..NOTYET  ;SET CARRY HERE
002C'  006E'                    .WORD DEXIT     ;$NULL
002E'  0063'                    .WORD ..NOGO    ;$IVAL
0030'  0063'                    .WORD ..NOGO    ;$FVAL
0032'  005B'                    .WORD ..STR     ;$STRADR
```

```
0034'  0044'                .WORD ..NAME     ;$NAMADR
0036'  004E'                .WORD ..INT      ;$IADR
0038'  004E'                .WORD ..INT      ;$FADR
003A'  0053'                .WORD ..ARA      ;$AADR
003C'  0063'                .WORD ..NOTYET   ;$CPLADR
003E'  0063'                .WORD ..NOTYET   ;$PIX
0040'  0063'                .WORD ..NOTYET   ;$GROUP
0042'  0063'                .WORD ..NOGO     ;$CMDADR

0044'  E5           ..NAME: PUSH H
0045'  CD 0000:15           CALL NXTVAL
0048'  2803                 JRZ ..DONT
004A'  CD 0019'              CALL DELBAG      ;DELETE VALUE
004D'  E1           ..DONT: POP H
004E'  CD 0071'     ..INT:  CALL CLEARVAL
0051'  181B                 JMPR DEXIT

0053'  CD 0067'     ..ARA:  CALL    DECUSE
0056'  CD 0000:04           CALL    AFREE
0059'  1813                 JMPR    DEXIT


005B'  CD 0067'     ..STR:  CALL DECUSE      ;DECR USE COUNT
005E'  CD 0000:1B           CALL SFREE
0061'  180B                 JMPR DEXIT
0063'                ..NOGO:
0063'                ..NOTYET:       ERROR ER.IMP[
0063'  CD 0000:0B  +         CALL ERRPGM
0066'  03          +         .BYTE ER.IMP
                   +]

0067'  E5           DECUSE: PUSH H   ;DECREMENT USE COUNT
0068'  C5                   PUSH B   ;ASSUMES USECOUNT #0
0069'  01 0005             LXI B,$USE
006C'  09                   DAD B
006D'  35                   DCR M
006E'  C1           DEXIT:  POP B
006F'  E1                   POP H
0070'  C9                   RET

                    ; *CLEARVAL--CLEARS 4 BYTES OF VALUE & SETS $NULL
                    ;IN:     HL->FWA OF NAME, IADR OR FADR
                    ;OUT:    HL SAME, FIRST BYTE=$NULL AND $NVALUE
                    ;        BYTES (4) ARE CLEAR

0071'  E5           CLEARVAL:       PUSH H
0072'  C5                   PUSH B
0073'  3602                 MVI M,$NULL      ;SET $NULL IN
0075'  01 0005              LXI B,$NVALUE    ;POINT TO WHERE TO CLEAR
0078'  09                   DAD B
0079'  01 0004              LXI B,4          ;CLEAR 4 BYTES
007C'  CD 0000:0A           CALL CLEARIT     ;GUESS WHAT
007F'  18ED                 JMPR DEXIT       ;BYE
```

```
                        ; *ISNAME-- SEES IF THING IS NAME OR NUMBER*
0081'   7E              ISNAME: MOV A,M
0082'   FE0A                   CPI $NAMADR
0084'   C8                     RZ
0085'   FE0C                   CPI $IADR
0087'   C8                     RZ
0088'   FE0E                   CPI $FADR
008A'   C9                     RET
                        ; *PSCAN--SCANS OVER BALANCED PARENS

008B'   D5              PSCAN:  PUSH D
008C'   1628                   MVI D,'('
008E'   1E29                   MVI E,')'
0090'   23                     INX H           ;POINT PAST '('
0091'   CD 0000:06             CALL BALLY
0094'   D1                     POP D
0095'   C9                     RET

                        ; *NAMSET--POINTS HL AT END OF NAME, DE AT BEGINNING*

                        ;IN     HL->NAME IN ASCII IN COMMAND, ETC
                        ;OUT    DE->NAME, HL->TERMINATOR OF NAME

0096'   E5              NAMSET: PUSH H
0097'   D1                     POP D
0098'   CD 0000:08      ..NAMS: CALL CARTYP     ;DELIM?
009B'   C8                     RZ              ;YUP
009C'   23                     INX H           ;NEXT CHAR
009D'   18F9                   JMPR ..NAMS


                        ;NAMADR--GETS NAME ADDRESS FOR YA
                        ;IN     DE->FIRST CHAR OF NAME
                        ;       HL->TERMINATOR AFTER NAME
                        ;               (SETUP BY NAMSET, PROBABLY)
                        ;       A=0 TO SEARCH SYSTEM NAMES,
                        ;       A=1 TO HASH OFF FIRST LETTER
                        ;OUT    HL->DELIM FOLLOWING NAME INCL SWITCH IF CMD
                        ;       DE->FWA OF NAME BLOCK IF Z SET
                        ;       OR SAME AS ENTRY IF NAME NOT FOUND,
                        ;          BUT Z CLEAR
                        ;       BC->POINTER TO CODE TO EXEC IF CMD
                        ;           OR AT $NVALUE BYTES IN NAMBLOCK
                        ;           OR, IF NOT FOUND (Z CLEAR CASE),
                        ;           BC->LAST NAME IN HASH TABLE
                        ;           CHAIN (THE ONE WITH 0 NLINK FIELD)
                        ;       BC+2->POINTER TO ARG LIST OR 0 FOR HELP
                        ;       IF STUPID NAME (STARTS WITH DELIM
                        ;          OR NUMBER), CARRY SET, OTHERWISE
                        ;          CARRY CLEAR.

                        ;ALGORITHM:  COMPARE (DE)+ WITH CURRENT NAME.
                        ;               WHEN MATCH FAILS IF DE=HL,
                        ;               THEN GOT A MATCH
                        ;               (IT WAS THE DELIMS THAT
```

```
                        ;                FAILED TO MATCH)
                        ;               OTHERWISE, NO MATCH, CHAIN ON
                        ;               ONE TOUGH CASE--IF HL->NULL,
                        ;               IT WILL MATCH NULL AT END
                        ;               OF NAME, SO WE COUNT HL->NULL
                        ;               AS A MATCH
                        ;               UNLESS, OF COURSE,
                        ;               HL=DE IN THE BEGINNING (ERROR)

009F'  DDE5             NAMADR: PUSH X
00A1'  FDE5                     PUSH Y
00A3'  E5                       PUSH H
00A4'  D5                       PUSH D          ;SAVE IF NO FIND
00A5'  DDE1                     POP X           ;I IX
00A7'  4F                       MOV C,A         ;SAVE FOR BELOW
00A8'  1A                       LDAX D          ;GET FIRST CHAR
00A9'  CD 0000:07               CALL CARTYA     ;CHECK IT
00AC'  385D                     JRC ..EXIT      ;NO PLACE FOR NUMBER
00AE'  37               ..DUMB: STC             ;SET IN CASE Z SET
00AF'  285A                     JRZ ..EXIT      ;NO DELIMS HERE EITHER
00B1'  13                       INX D           ;CHECK SECOND
00B2'  1A                       LDAX D          ;...
00B3'  CD 0000:07               CALL CARTYA     ;IF DELIM, IS A-Z
00B6'  1B                       DCX D           ;BACKUP
00B7'  F5                       PUSH PSW        ;SAVE CC'S & A
00B8'  1A                       LDAX D          ;GET FIRST CHAR AGAIN
00B9'  CD 0000'                 CALL FINDHASH   ;GET FWA OF TABLE
00BC'  79                       MOV A,C         ;0/1 FROM ENTRY& "MOV C,A"
00BD'  B7                       ORA A           ;IS SYSNAMES (=0)?
00BE'  2806                     JRZ ..SYSN      ;YUP
00C0'  F1                       POP PSW         ;CHECK CC'S FROM CHKAZ
00C1'  200A                     JRNZ ..NAML     ;IS AT LEAST 2 CHARS
00C3'  EB                       XCHG            ;DE HAS POINTER NOW
00C4'  1845                     JMPR ..EXIT
00C6'  F1               ..SYSN: POP PSW         ;GET RID OF PUSHEE
00C7'  01 000E                  LXI B,14        ;POINT TO SYSNAMES
00CA'  CD 0000:13               CALL NXTABC
00CD'  E5               ..NAML: PUSH H          ;SAVE PTR IN IY
00CE'  FDE1                     POP Y
00D0'  01 0009                  LXI B,$NASCII   ;POINT AT ASCII
00D3'  09                       DAD B
00D4'  1A               ..NAR:  LDAX D          ;LOAD CHAR INTO A
00D5'  BE                       CMP M           ;SAME AS (HL)?
00D6'  2023                     JRNZ ..NAG      ;NOPE
00D8'  AF                       XRA A           ;IS (HL)=0
00D9'  BE                       CMP M           ;???
00DA'  280C                     JRZ ..NAM       ;YUP--MATCH
00DC'  23                       INX H           ;BUMP
00DD'  13                       INX D           ;BUMP
00DE'  C1                       POP B           ;TERMINATOR ADDR IN BC
00DF'  C5                       PUSH B          ;AND PUT BACK
00E0'  79                       MOV A,C         ;CHECK FOR =S
00E1'  BB                       CMP E           ;?
00E2'  20F0                     JRNZ ..NAR      ;LOOK MORE
00E4'  78                       MOV A,B
```

```
00E5'   BA                      CMP D
00E6'   20EC                    JRNZ ..NAR
00E8'   FDE5            ..NAM:  PUSH Y
00EA'   D1                      POP D               ;Z=1, DE->FWA NAMEBLOCK
00EB'   62                      MOV H,D             ;COPY TO HL
00EC'   6B                      MOV L,E
00ED'   01 0005                 LXI B,$NVALUE       ;WANT BC->NVALUE
00F0'   09                      DAD B
00F1'   44                      MOV B,H
00F2'   4D                      MOV C,L             ;FOR EVERYBODY
00F3'   1A                      LDAX D              ;GET TYPE OF NAME
00F4'   FE18                    CPI $CMDADR         ;IF CMD, CHECK SWITCH
00F6'   2819                    JRZ ..SWIT          ;DO IT
00F8'   AF              ..COUT: XRA A               ;CLEAR THAT CARRY
00F9'   1810                    JMPR ..EXIT
00FB'   FDE5            ..NAG:  PUSH Y
00FD'   E1                      POP H
00FE'   DDE5                    PUSH X
0100'   D1                      POP D               ;RESTORE DE
0101'   CD 0000:14              CALL NXTLNK         ;GET THAT LINK
0104'   20C7                    JRNZ ..NAML         ;IF NON-ZERO
0106'   FDE5                    PUSH Y              ;GET LAST ONE IN BC
0108'   C1                      POP B               ;LIKE THAT
0109'   F6FF                    ORI -1              ;CLEAR Z
010B'   E1              ..EXIT: POP H               ;GET ORIG TERMY BACK
010C'   FDE1            ..EX1:  POP Y
010E'   DDE1                    POP X
0110'   C9                      RET
0111'   E1              ..SWIT: POP H               ;GET ORIG TERMY BACK
0112'   E5                      PUSH H              ;FOR ..COUT
0113'   7E                      MOV A,M             ;CHECK DELIM FOR SWITCH
0114'   FE2E                    CPI EXTDEL
0116'   20E0                    JRNZ ..COUT
0118'   E1                      POP H               ;STACK MANAGEMENT
0119'   23                      INX H               ;POINT PAST
011A'   CD 0000:08              CALL CARTYP         ;ALPHANS ONLY
011D'   288F                    JRZ ..DUMB          ;DUMMY!
011F'   03                      INX B               ;JUMP OVER 4BYTES INFO
0120'   03                      INX B
0121'   03                      INX B
0122'   03                      INX B
0123'   0A              ..SCMR: LDAX B              ;SCAN TO Z OF
0124'   03                      INX B               ;ASCIZ NAME
0125'   B7                      ORA A
0126'   20FB                    JRNZ ..SCMR         ;PAST NULL NOW
0128'   0A                      LDAX B              ;DOUBLE NULL?
0129'   B7                      ORA A
012A'                   ..BAD:  ZERROR ER.SW        ;SWITCH ERROR!
012A'   2004            +       JRNZ ..0001
012C'   CD 0000:0B      +       CALL ERRPGM
012F'   32               +       .BYTE ER.SW
0130'                    +..0001:]
0130'   C5                      PUSH B              ;SAVE B
0131'   E5                      PUSH H              ;SAVE PTR TO FIRST CHAR
0132'   03                      INX B               ;OF TYPED SWITCH
```

```
0133'   03                      INX B           ;NOW ADVANCE PAST
0134'   03                      INX B           ;ACODE AND
0135'   03                      INX B           ;ARGLIST PTRS
0136'   7E              ..SWIC: MOV A,M         ;GET USER CHARACTER
0137'   B7                      ORA A           ;IS NULL?
0138'   2810                    JRZ ..OKSW      ;MATCH!
013A'   0A                      LDAX B          ;GET SWITCH CHAR
013B'   BE                      CMP M           ;SEE IF MATCHES
013C'   23                      INX H           ;BUMP ANYWAY
013D'   03                      INX B           ;B TOO
013E'   28F6                    JRZ ..SWIC      ;MATCHES!
0140'   2B                      DCX H           ;POINT BACK AT DELIM
0141'   CD 0000:08              CALL CARTYP     ;CHECK IF USER AT DELIM
0144'   2804                    JRZ ..OKSW      ;ONLY WAY TO GET OUT
0146'   E1                      POP H           ;RESTORE PTR TO TYPED SW
0147'   F1                      POP PSW         ;TRASH OLD BC, USE NEW
0148'   18D9                    JMPR ..SCMR     ;SCAN MORE
014A'   F1              ..OKSW: POP PSW         ;TRASH OLD HL
014B'   C1                      POP B           ;GET ->TO CODE&ARGLIST
014C'   AF                      XRA A           ;SUCCESS
014D'   18BD                    JMPR ..EX1      ;BYE!

                        ; *NAMBLD--BUILDS A NAME NODE*
                        ;IN:    DE&HL&BC SETUP BY NAMADR
                        ;OUT    HL SAME
                        ;       DE->FWA OF NAMEBLOCK
                        ;       BC CLOBBERED

                        ;NAMEBLOCK LOOKS LIKE:  TYPE
                        ;                       LENGTH
                        ;                       FLAGS
                        ;                       LINK TO NEXT NAME(2)
                        ;                       LINK TO VALUE(2) OR;
                                  INTEGER VALUE
                        ;                       FLOAT VALUE OR ZERO(2)

                        ;                       ASCIZ NAME

                        ;GAME PLAN:     1. FILL IN TYPE AND FLAGS
                        ;               2. CHAIN IN $NLINK
                        ;               3. COPY IN ASCIZ NAME

014F'   DDE5            NAMBLD: PUSH X
0151'   E5                      PUSH H
0152'   C5                      PUSH B
0153'   A7                      ANA A           ;CLEAR CARRY
0154'   ED52                    DSBC D          ;GET LENGTH
0156'   E5                      PUSH H          ;IN HL
0157'   7D                      MOV A,L         ;GET LENGTH/16
0158'   C609                    ADI $NASCII     ;ADD IN HEADER LENGTH
015A'   E6F0                    ANI 0F0H        ;CLEAR LOW BITS
015C'   1F                      RAR
015D'   1F                      RAR
015E'   1F                      RAR
015F'   1F                      RAR             ;DIV BY 16
```

```
0160'   CD 0000:05             CALL ALLOC          ;WANT A BLOX
0163'   E5                     PUSH H              ;HL HAS FWA OF BLOCK
0164'   3602                   MVI M,$NULL         ;VALUELESS NAME
0166'   23                     INX H               ;POINT TO FLAGS
0167'   23                     INX H               ;AND CLEAR FROM THERE ON
0168'   01 0007                LXI B,$NASCII-2
016B'   CD 0000:0A             CALL CLEARIT        ;ZEROES
016E'   E1                     POP H               ;GET FWA FROM ALLOC ABOVE
016F'   C1                     POP B               ;GET LENGTH FROM DSBC ABOVE
0170'   DDE1                   POP X               ;GET BC FROM ENTRY (LAST
                                                    ;IN HASH TABLE ADDR)
0172'   DD7503                 MOV $NLINK(X),L     ;POINT HASH ENTRY
0175'   DD7404                 MOV $NLINK+1(X),H       ;TO US
0178'   E5                     PUSH H              ;SAVE FWA AGAIN
0179'   C5                     PUSH B              ;SAVE LENGTH
017A'   01 0009                LXI B,$NASCII       ;POINT AT ASCIZ LOC
017D'   09                     DAD B               ;HL->WHERE TO STICK
017E'   C1                     POP B               ;RESTORE B
017F'   EB                     XCHG                ;DE HAS WHERE FROM
0180'   EDB0                   LDIR                ;DO IT
0182'   AF                     XRA A
0183'   12                     STAX D              ;STUFF A NULL
0184'   D1                     POP D               ;FWA IN DE
0185'   E1             ..BYE:  POP H
0186'   DDE1                   POP X
0188'   C9                     RET                 ;BYE

                        ;ARG
                        ;
                        ;       IN:     HL->TERMINATOR (BLANK, COMMA)
                        ;               CALL    ARG
                        ;               .BYTE   ARG1,ARG2,...,$TAF
                        ;
                        ;       OUT:    IY->TYPE
                        ;                   STUFF
                        ;                   TYPE
                        ;                   STUFF
                        ;                   ...
                        ;                   0
                        ;               AND HL->TERMINATOR
                        ;               (BLANK, COMMA, NL OR NULL)


                        ;NARRATIVE BY JAY FENTON:

                        ;ZGRASS ARGUMENT SCANNER
                        ;THIS ROUTINE PREPROCESSES THE ARGUMENTS TO A
                        ;COMMAND AND LEAVES THEM ON THE VALUE STACK
                        ;(IY STACK).  A LIST SPECIFYING HOW TO
                        ;SCAN THE ARGUMENTS FOLLOWS INLINE AFTER THE
                        ;CALL
                        ;ON ENTRY, HL POINTS AT THE DELIMITER FOLLOWING
                        ;THE COMMAND NAME

                        ;ON EXIT, HL POINTS BEYOND THE LAST ARG SCANNED
```

```
                        ;AND IY POINTS AT THE FIRST ENTRY ON THE
                        ;VALUE STACK.
                        ;ENTRIES ON THE STACK CONSIST OF A TYPE BYTE
                        ;FOLLOWED BY 4 BYTES OF VALUE

                        ;REGISTER USE: HL=STRING POINTER
                        ;                IX=LIST POINTER
                        ;                IY=VALUE POINTER

0189'  DDE3             ARG:    XTIX    ;SAVE IX AND USE TO FETCH ARGS
018B'  CD 0000:0F       ..ARG1: CALL    INC5IY
018E'  FDE5                     PUSH Y          ;SAVE FOR LATER
0190'  CD 0000:11       ..ARG2: CALL KISNL      ;EOL ALREADY?
0193'  282E                     JRZ ..EMTY      ;YUP
0195'  FE29                     CPI ')'         ;CLOSE PAREN?
0197'  282A                     JRZ ..EMTY      ;OK
0199'  FE2C                     CPI ','         ;COMMA?
019B'  2001                     JRNZ ..ARG3     ;MUSTA BEEN SPACE
019D'  23                       INX H           ;SKIP COMMA
019E'  DD7E00           ..ARG3: MOV A,0(X)      ;GET TYPE PANTED
01A1'  FE00                     CPI $TAF        ;THAT'S ALL?
01A3'  200B                     JRNZ ..ARG4     ;NAH
01A5'  FD360000         ..AOUT: MVI 0(Y),0      ;TERMINATE LIST
01A9'  FDE1                     POP Y           ;WIND DOWN
01AB'  DD23                     INX X           ;JUMP OVER $TAF,ETC.
01AD'  DDE3                     XTIX            ;RSTORE X
01AF'  C9                       RET

01B0'  FE1E             ..ARG4: CPI $REPEAT     ;MORE?
01B2'  2005                     JRNZ ..ARG5     ;NO, NEW TYPE
01B4'  DD2B                     DCX X           ;BACKUP
01B6'  DD7E00                   MOV A,0(X)      ;GET PREVIOUS TYPE
01B9'  CD 0000:0C       ..ARG5: CALL EVAL       ;DO FETCH
01BC'  CD 0000:0F               CALL INC5IY     ;SPACE TO NEXT
01BF'  DD23                     INX X
01C1'  18CD                     JMPR ..ARG2     ;DO IT AGAIN

01C3'  DD7E00           ..EMTY: MOV A,0(X)      ;SEE IF $TAF
01C6'  FE00                     CPI $TAF        ;IF SO, EXIT
01C8'  28DB                     JRZ ..AOUT
01CA'  FE1E                     CPI $REPEAT     ;IF REPEAT, DITTO
01CC'  28D7                     JRZ ..AOUT
01CE'  DD23                     INX X           ;TRY 1 PAST
01D0'  DD7E00                   MOV A,0(X)      ;SEE IF 1 PAST IS
01D3'  FE1E                     CPI $REPEAT     ;IS REPEAT
01D5'  28CE                     JRZ ..AOUT      ;ALSO OK (NO ARGS CASE)
                                ERROR ER.UNF[
01D7'  CD 0000:0B       +       CALL ERRPGM
01DA'  27               +       .BYTE ER.UNF
                        ]


01DB'  FD7E00           PRTYPE: MOV     A,0(Y)  ;GET TYPE
                        ;+
                        ; PRTYPA - PRINTS TYPE AS PASSED IN A
```

```
                        ;-
01DE'   C5              PRTYPA: PUSH B
01DF'   E5                      PUSH H          ;PRINT TYPE WANTED
01E0'   4F                      MOV C,A         ;MAKE BC OFFSET
01E1'   0600                    MVI B,0
01E3'   21 01FA'                LXI H,..PTAB    ;GET BASE
01E6'   CD 0000:13              CALL NXTABC     ;GET ADDRESS
01E9'   CD 0000:19              CALL PRINTHL    ;PRINT IT
01EC'   BE                      CMP M           ;IS DOUBLE NULL?
01ED'   2008                    JRNZ ..DONE     ;NAH
01EF'   CD 0000:18              CALL PRINTFF
01F2'   4E414D4500              .ASCIZ /NAME/
01F7'   E1              ..DONE: POP H           ;POP
01F8'   C1                      POP B
01F9'   C9                      RET

01FA'   020C'           ..PTAB: .WORD ..LOST    ;$TAF
01FC'   022C'                   .WORD ..NAME    ;$NULL
01FE'   0217'                   .WORD ..NUM     ;$IVAL
0200'   0217'                   .WORD ..NUM     ;$FVAL
0202'   0210'                   .WORD ..STR     ;$STRADR
0204'   022C'                   .WORD ..NAME    ;$NAMADR
0206'   0225'                   .WORD ..NUMN    ;$IADR
0208'   0225'                   .WORD ..NUMN    ;$FADR
020A'   021E'                   .WORD ..ARRN    ;$AADR


020C'   3F3F3F00        ..LOST: .ASCIZ /???/
0210'   535452494E47    ..STR:  .ASCIZ  /STRING/
0217'   4E554D424552    ..NUM:  .ASCIZ /NUMBER/
021E'   415252415900    ..ARRN: .ASCIZ /ARRAY/
0224'   00                      .BYTE 0
0225'   4E554D424552    ..NUMN: .ASCIZ /NUMBER/
022C'   0000            ..NAME: .BYTE 0,0
022E'                   NAMEND: .END
```

NAM - ZGRASS NAME ROUTINES
+++++ SYMBOL TABLE +++++

```
AASN    005F          AFREE   0000:04 X     ALLOC   0000:05 X     ARG     0189'   I
ARGSTK  60CC          BACKGR  65CC          BALLY   0000:06 X     BLANK   0020
BOTRAM  6000          BOTTOM  64A9          CARTYA  0000:07 X     CARTYP  0000:08 X
CHARSL  65DD          CHKAZ   0000:09 X     CLEARI  0000:0A X     CLEARS  60CA
CLEARV  0071'   I     CNTRL   000C          CNTRLC  65CD          CNTRLO  65D9
CNTRLU  0015          CNTRLZ  65E4          CPLARE  611C          CPLSIZ  0140
CR      000D          CSBLOK  668D          CSFLAG  6260          CURCX   6814
CURCY   6816          CURREN  64A5          C.CO    0011          C.C1    0012
C.CX    000B          C.CY    000D          C.DP    0013          C.ST    0002
C.X     0003          C.XF    000F          C.XS    0007          C.Y     0005
C.YF    0010          C.YS    0009          DDTON   65CE          DECUSE  0067'
DELBAG  0019'   I     DEVBL   6589          DEVCL0  6579          DEVCL1  657B
DEVCL2  657D          DEVCL3  657F          DEVCL4  6581          DEVCL5  6583
DEVCL6  6585          DEVCL7  6587          DEVFB   65CB          DEVHCB  658F
DEVMO   658B          DEVNM   65B7          DEVNT   658D          DEVTNA  65BB
DEVTNB  65BF          DEVTNC  65C3          DEVVA   65BD          DEVVAR  6579
DEVVB   65C1          DEVVBL  6591          DEVVC   65C5          DEVVD   65C9
DEVVN   65B9          DEVVS   ω       DEVXCD 65B3         DEVYCD 65B5
DEXIT   006E'         DOLPLH  62E8          DOLPPT  62EA          DUMBST  6577
EDBCNT  64AD          EDCCNT  64A7          EDLONG  6812          EDMODE  681C
EDNAME  64A1          EDNCX   680C          EDNCY   680E          EDNEWS  64A5
EDOCX   6808          EDOCY   680A          EDPN    6806          EDPO    6804
EDPTRC  64AB          EDPTRL  64A9          EDSTR   681A          ERABIT  0002
ERRPGM  0000:0B X     ER.ARA  002F          ER.ARG  0034          ER.ASN  0015
ER.BOX  001A          ER.CHN  0002          ER.CMD  001F          ER.CNV  0016
ER.COR  001B          ER.CTL  0036          ER.DEL  0026          ER.DIM  0030
ER.DIV  0018          ER.DP   0037          ER.DSK  0019          ER.EDT  0035
ER.FMT  0038          ER.FNF  001C          ER.FOR  0028          ER.IMP  0003
ER.LAB  0025          ER.MAC  0022          ER.NAE  002D          ER.NAM  0029
ER.NEG  003A          ER.NOT  0023          ER.NUL  0039          ER.NUM  002B
ER.NXT  001D          ER.OFL  0017          ER.OPN  0014          ER.OVE  001E
ER.PAR  002A          ER.REN  002C          ER.RET  0024          ER.SEP  0021
ER.SNP  0031          ER.SPC  002E          ER.STK  0004          ER.SW   0032
ER.TER  0020          ER.UFL  0033          ER.UNF  0027          EVAL    0000:0C X
EXTDEL  002E          E.HVAL  0002          E.LVAL  0001          E.SIZ   0005
E.TYP   0000          E.VAL   0001          FCNTH   65D2          FCNTI   65D3
FCNTJ   65D4          FCNTK   65D5          FCNTL   65D6          FCNTV   65E0
FCNTY   65E3          FINDHA  0000'   I     FIRST   64A3          FLAGS   65CB
FOREGR  65D0          FRAGSI  0400          FREE    0000:0D X     FREELS  65E5
FSTDOL  648C          FSTINT  62EC          FWDPTR  65E7          GETLPT  0000:0E X
HCAREA  6593          INC5IY  0000:0F X     INCRO   65E9          INPTTY  0000:10 X
ISNAME  0081'   I     JUNK    6542          KBLOCK  65F1          KEYFLG  67FF
KEYPTK  6533          KEYTRK  67F7          KISNL   0000:11 X     LF      000A
LISTON  65E2          MACSTU  6536          MACTOP  625E          MAXFRG  0040
MNMX    65EB          NAMADR  009F'   I     NAMBLD  014F'   I     NAMEND  022E'
NAMSET  0096'   I     NBLKB   0000          NBLKM   0001          NEWBOT  6810
NL      000A          NLPNT   0000:12 X     NSADDR  6802          NUMBUF  64A3
NXTABC  0000:13 X     NXTLNK  0000:14 X     NXTVAL  0000:15 X     OLDCHR  64A0
OLDCUR  649F          OLDKEY  649D          OLDXY   65EF          ONEBUF  6729
OPRL    0014          OPRSP   655B          OPRSTK  6547          OPRSZ   655D
OUTCH   0000:16 X     OUTOFF  62E7          PCNT    655E          PIXVAL  65ED
POINTE  64A7          PONOFF  65DA          PRINTA  0000:17 X     PRINTF  0000:18 X
PRINTH  0000:19 X     PRINTR  6540          PRTYPA  01DE'   I     PRTYPE  01DB'
PSCAN   008B'   I     PUTOPN  0000:1A X     RAMEND  7FFF          RAMSTR  6900
RANSHT  655F          RMDTMP  6538          RUBACK  0005          RUBOUT  007F
```

NAM - ZGRASS NAME ROUTINES
+++++ SYMBOL TABLE +++++

SAVESP	625C	SCNEED	0004	SCRWIN	67CD	SFREE	0000:1B X
SOPRSP	653D	SOPRSZ	653F	STACK	60C8	STAKTO	6000
STRSIZ	6800	SUBSTU	6534	TAB	0009	TABPNT	0000:1C X
TAPBUF	64B3	TAPCON	64AF	TAPPRO	64B1	TBFEND	6533
TEMPCH	0000:1D X	TEMPHD	60CA	TEMPS	62E5	TEMPUN	0000:1E X
TMPARG	67AD	TOP	6818	TTYBEG	6261	TTYEND	62E1
TTYINT	62E3	TTYPTR	62E1	TXTWIN	67E2	UARTFL	649C
USREND	65F1	V3PTR	6573	VDCHAR	649E	VDNLF	65CF
VIPLH	6545	VOICE0	6563	WRMODE	65EE	ZGIM2	0001
ZGREND	681D	$AADR	0010	$ADDRF	0007	$ADDRI	0005
$ADDRS	0009	$ANY	001A	$ANYNA	FFFC	$ANYVA	FFFE
$ARGPT	0011	$BGPTR	000F	$BNDL	0007	$CALLE	000D
$CMDAD	0018	$CPLBL	002A	$CSBL0	0028	$DATAP	0007
$DOLDE	0001	$DOLOO	0002	$DVAL	0000	$END	001C
$FADR	000E	$FLAGS	0002	$FORBL	0024	$FORPT	000B
$FVAL	0006	$GOSUB	001A	$IADR	000C	$INPBU	0018
$INPPT	0016	$IVAL	0004	$KEYBL	0026	$LENGT	0001
$LINPT	0005	$LOCPT	0009	$MIBBL	0022	$MIBEN	001B
$NAMAD	000A	$NAME	000A	$NASCI	0009	$NDEL	0080
$NLINK	0003	$NULL	0002	$NVALU	0005	$REPEA	001E
$RVSTU	0013	$SAME	0020	$SASCI	000A	$SLEN	0006
$STRAD	0008	$STRPT	0003	$TAF	0000	$TYPE	0000
$USE	0005	.BLNK.	0000:03 X	.DATA.	0000* X	.PROG.	022E X

```
 .INSERT A:S.ASM
@.REMARK /
@                       *
@                     * * *
@                       ***
@                     *******
@                         *
@                        *
@                       *
@                      *
@                     *
@                     *******
@
@      "WHEN YOU CARE ENOUGH TO PROGRAM
@                 THE VERY BEST"
@
@              ZGRASS V2.00000000
@BY JAY FENTON, NOLA DONATO, AND TOM DEFANTI
@                    (C) 1978
@
@/
 .INSERT A:ZRAM.ASM
 .INSERT A:IOEQU.ASM
         .LINK
         .PREL
         .IDENT  .MAIN.
         .INTERN .MAIN.
         .INTERN ABORT
         .INTERN ARGLIN
         .INTERN CMDREN
         .INTERN CMDRES
         .INTERN CMDSKP
         .INTERN CMDSKO
         .INTERN CNTCK
         .INTERN CZCHEK
         .INTERN DOCMD
         .INTERN DOMORE
         .INTERN ERRPGM
         .INTERN INIDEV
         .INTERN LINEP
         .INTERN PRLINE
         .INTERN RESTART
         .INTERN RETNONE
         .INTERN RETONE
         .INTERN RGBINI
;
         .EXTERN ALLOC
         .EXTERN ARG
         .EXTERN ARVAL
         .EXTERN CARTYP
         .EXTERN CLEARIT
         .EXTERN CMDTAB
         .EXTERN CORCHK
         .EXTERN CORINI
         .EXTERN DELMIB
```

```
                        .EXTERN DOCHK
                        .EXTERN EVAL
                        .EXTERN EVALARG
                        .EXTERN FINDHASH
                        .EXTERN FREE
                        .EXTERN FREEALL
                        .EXTERN GETLPT
                        .EXTERN INIVDR
                        .EXTERN INPTTY
                        .EXTERN ISNL
                        .EXTERN ISTERM
                        .EXTERN KBMIB
                        .EXTERN KILMIB
                        .EXTERN KISNL
                        .EXTERN KLB
                        .EXTERN MAKMIB
                        .EXTERN NAMADR
                        .EXTERN NAMSET
                        .EXTERN NLPNT
                        .EXTERN NXTABC
                        .EXTERN NXTLNK
                        .EXTERN NXTPTR
                        .EXTERN NXTVAL
                        .EXTERN OUTCH
                        .EXTERN PDECA
                        .EXTERN PRINT
                        .EXTERN PRINTA
                        .EXTERN PRINTFF
                        .EXTERN PSCAN
                        .EXTERN PUTLPT
                        .EXTERN SCANLAB
                        .EXTERN SCANNL
                        .EXTERN SCLEAR
                        .EXTERN SWITCH
                        .EXTERN UARTINT
                        .EXTERN ZGINT

0000'   00              ZBEGIN: NOP
0001'   C3 0004'                JMP     ZBEGIN
0004'   AF              ZBEGIN: XRA     A
0005'   D320                    OUT     NORML
0007'   C3 03E8'                JMP RESTART
000A'                           .BLKW   3
                        S
0010'   0000                    .WORD   0
                        DR
0012'   0000          INTTAB:   .WORD   0
0014'   0000:2F                 .WORD   UARTINT
0016'   0000:30        ZGITAB:   .WORD   ZGINT
0018'                           .BLKB   4EH-26
004C'                   INIDEV:
004C'   0000                    .WORD   0
004E'   007B                    .WORD   123
0050'   00B9                    .WORD   185
0052'   00FB                    .WORD   251
```

; ALIGN LIGHT PEN GOODIE

; LIGHT PEN INTERRUPT AD

; NCU SERVICE REQUEST
; UART INTERRUPT

76-26-24
26/50
(26)

ALLOC ✓
ERRPGM ✓
GETCH
HEXBYT
INCSIY
INCUSE
NXTVAL ✓
OUTCH ✓

```
0054'   0007                    .WORD   7
0056'   005B                    .WORD   5BH
0058'   00A5                    .WORD   0A5H
005A'   0008                    .WORD   8
005C'   0001                    .WORD   1
005E'   0047                    .WORD   47H
0060'   0003                    .WORD   3
0062'   002C                    .WORD   44
0064'   00CC                    .WORD   204
0066'                   ENDDEV:
                        ; ZGRASS NMI ROUTINE
0066'   00                      NOP
0067'   C3 006A'                JMP     NMIROU
006A'   AF              NMIROU: XRA     A
006B'   D320                    OUT     NORML
006D'   C3 03E8'                JMP     RESTART
0070'   C3 00A2'        RESDDT: JMP     ABORT

                        ;********Z*G*R*A*S*S*****USER COMMAND ACTION****

0073'   3A 65CD         CNTCK:  LDA CNTRLC      ;CHECK ^C
0076'   B7                      ORA A           ;IS ZILCH?
0077'   2029                    JRNZ ABORT      ;NOPE-ABORT
0079'   3A 65E4                 LDA CNTRLZ      ;GET CONTROL Z
007C'   B7                      ORA A
007D'   C9                      RET             ;SET CC'S ONLY
007E'   7C              KILIST: MOV A,H         ;IGNORE IF ZERO
007F'   B5                      ORA L
0080'   C8                      RZ
0081'   CD 0000:0C      ..K1:   CALL DELMIB     ;ZAP MIB
0084'   01 000F                 LXI B,$BGPTR    ;CHAIN TO NEXT
0087'   CD 0000:20              CALL NXTABC     ;...
008A'   20F5                    JRNZ ..K1
008C'   C9                      RET


008D'   2A 625E          ZAPCZ: LHLD MACTOP     ;CHECK CONTRLZ
0090'   3A 6260                 LDA CSFLAG      ;MIB LIST FIRST
0093'   B7                      ORA A           ;IS ANY MAYBE?
0094'   C8                      RZ              ;NAH
0095'   CD 0000:19              CALL KILMIB     ;ZAP THEM
0098'   21 669A                 LXI H,CSBLOK+$CALLER    ;POINT AT OLD
009B'   CD 0000:22              CALL NXTPTR     ;MACTOP THAT CONTROLZEED
009E'   22 625E                 SHLD MACTOP     ;AND STORE IT
00A1'   C9                      RET


00A2'   31 60C8          ABORT: LXI SP,STACK    ;RESET STACK
00A5'   2A 62E8          ..KILL:LHLD DOLPLH     ;ZAP BACKGROUND
00A8'   CD 007E'                CALL KILIST
00AB'   2A 6545                 LHLD VIPLH      ;FOREGROUND TOO
00AE'   CD 007E'                CALL KILIST
00B1'   22 62E8                 SHLD DOLPLH     ;ZERO POINTER
00B4'   22 6545                 SHLD VIPLH      ;THIS ONE TOO
00B7'   CD 008D'                CALL ZAPCZ      ;ZAP MIB CHAIN ON CONTRLZ
```

```
00BA'   CD 0000:19          CALL KILMIB     ;ZAP MIB'S AND CALLERS
00BD'   21 0000:24          LXI     H,OUTCH
00C0'   22 6540             SHLD    PRINTR
00C3'   AF              ..K3:   XRA A           ;NULL OUT CNTRLC
00C4'   32 65CD             STA CNTRLC      ;...
00C7'   32 65D9             STA CNTRLO      ;TURN ON ALL PRINT, ETC.
00CA'   32 65E4             STA CNTRLZ      ;CNTRLZ TOO
00CD'   32 6260             STA CSFLAG      ;CSFLAG TOO
00D0'   32 65CE             STA DDTON       ;SINGLE STEP
00D3'   32 65DA             STA PONOFF      ;PRINT WAIT
00D6'   32 65E2             STA LISTON      ;CONTROL X TOO
00D9'   32 655E             STA PCNT        ;FOR NOLA'S CODE
00DC'   3E01                MVI     A,1
00DE'   32 649C             STA     UARTFL
00E1'   3A 649C             LDA     UARTFL
00E4'   A7                  ANA     A
00E5'   1809                JMPR    ..OK
00E7'   2807                JRZ     ..OK
00E9'   AF                  XRA     A
00EA'   32 649C             STA     UARTFL
00ED'   CD 0468'            CALL    INUART
00F0'   21 6261         ..OK:   LXI H,TTYBEG    ;SETUP TTY BUFFER
00F3'   22 62E1             SHLD TTYPTR
00F6'   22 62E3             SHLD TTYINT
00F9'   CD 0000:28          CALL PRINTFF
00FC'   5E43                .ASCIZ /^C
00FE'   0D0A00          /
0101'   FB                  EI
0102'   21 60CC        ZGRASS: LXI H,ARGSTK    ;LOADUP IY
0105'   E5                  PUSH H
0106'   FDE1                POP Y
0108'   21 0000            LXI H,0         ;CLEAR MACTOP
010B'   22 625E             SHLD MACTOP     ;FOR KBMIB
010E'   21 002A             LXI     H,'*'
0111'   22 6577             SHLD    DUMBST
0114'   3A 65CB             LDA     DEVFB
0117'   A7                  ANA     A
0118'   2806                JRZ     ..NOFB
011A'   21 6577             LXI     H,DUMBST
011D'   22 6573             SHLD    V3PTR
0120'   CD 0000:28      ..NOFB: CALL PRINTFF
0123'   3E00                .BYTE   '>',0   ; FORGET THE BEEP
0125'   2A 60CA             LHLD TEMPHDR    ;ZAP ALL TEMPS
0128'   CD 0000:12          CALL FREEALL
012B'   21 0000            LXI H,0         ;NULL LIST HEADER
012E'   22 60CA             SHLD TEMPHDR
0131'   CD 01A5'            CALL DNULLS     ;ZAP ALL $NULL NAMES
0134'   CD 0139'            CALL ZLINEP     ;WAIT FER LINE
0137'   18C9                JMPR ZGRASS     ;THAT'S ALL
0139'   21 65F1        ZLINEP: LXI H,KBLOCK    ;KBM MIB&BUFFER
013C'   CD 0000:18          CALL KBMIB
013F'   21 660C             LXI H,KBLOCK+$MIBEND    ;SET BUFFER ADDR
0142'   CD 0073'        ZLOOP:  CALL CNTCK      ;CHECK ^C
0145'   CD 0000:15          CALL INPTTY     ;GET A LINE
0148'   E5                  PUSH    H
```

```
0149'   F5                      PUSH    PSW
014A'   CD 0162'                CALL STUFFCHK   ;SEE IF KEYPAD/CONTROLS
014D'   3A 65CC                 LDA BACKGR      ;BACKGROUND OFF (^B)
0150'   B7                      ORA A
0151'   CC 0000:0D              CZ      DOCHK   ; DO ANY DOLOOP STUFF
0154'   F1                      POP     PSW
0155'   E1                      POP     H
0156'   20EA                    JRNZ ZLOOP      ;IF NOT THERE, WAIT
0158'   CD 01DA'        ZGRIN:  CALL LINEP      ;PROCESS IT
015B'   CD 0000:17              CALL ISTERM     ;CHECK TERMY
015E'   B7                      ORA A           ;RET IF NULL
015F'   20F7                    JRNZ ZGRIN      ;ELSE PROCEED
0161'   C8                      RZ              ;RETURN

                        ; *STUFFCHK     ;CHECKS IF MACRO OR SUBR PENDING
                                        ;FROM KEYPAD OR CONTROL COMMAND
0162'   E5              STUFFCHK:      PUSH H
0163'   D5                      PUSH D
0164'   21 6534                 LXI H,SUBSTUFF
0167'   CD 0000:22              CALL NXTPTR
016A'   280C                    JRZ ..MAC       ;NUTTIN TO DO
016C'   01 0000                 LXI B,0
016F'   ED43 6534               SBCD SUBSTUFF
0173'   01 0178'                LXI B,..MAC
0176'   C5                      PUSH B          ;RET ADDR
0177'   E9                      PCHL            ;GO DOIT
0178'   21 6536          ..MAC: LXI H,MACSTUFF  ;ANY MACRO TO DO
017B'   CD 0000:22              CALL NXTPTR
017E'   C4 0238'                CNZ DOCMD       ;WELL, DO IT THEN!
0181'   D1                      POP D
0182'   E1                      POP H
0183'   C9                      RET
                        ; *PRCHEK-CHECKS IF CURRENT LINE NEEDS PRLINE
0184'   3A 65E2          PRCHEK: LDA LISTON
0187'   B7                      ORA A
0188'   200D                    JRNZ PRLINE     ;DO IT
018A'   3A 65CE                 LDA DDTON
018D'   B7                      ORA A           ;DEBUG ON?
018E'   C8                      RZ
018F'   ED4B 625E               LBCD MACTOP     ;ONLY IF MACRO
0193'   0A                      LDAX B
0194'   FE22                    CPI $MIBBLK
0196'   C0                      RNZ             ;SKIP IT

                        ; *PRLINE-PRINTS CURRENT LINE OUT*
                        ;       A.K.A. ARGLIN

0197'                   ARGLIN:
0197'   E5              PRLINE: PUSH H
0198'   CD 0000:13              CALL GETLPT     ;GET THAT LINE
019B'   3E0A                    MVI A,NL        ;PRINT TILL NL OR NULL
019D'   CD 0000:27              CALL PRINTA
01A0'   CD 0000:1F              CALL NLPNT
01A3'   E1                      POP H
01A4'   C9                      RET
```

```
                        ; *DNULLS--ZAPS ALL $NULL NAMES*

01A5'   3E41            DNULLS: MVI A,'A'
01A7'   F5              ..NCHR: PUSH PSW          ;SAVE CURRENT CHAR
01A8'   CD 0000:10              CALL FINDHASH     ;GET FWA
01AB'   E5              ..NEXL: PUSH H            ;SAVE POINTER IN IX
01AC'   DDE1                    POP X
01AE'   CD 0000:21      ..NEXH: CALL NXTLNK       ;GET NEXT
01B1'   281A                    JRZ ..NEXA        ;NONE-TRY NEXT LETTER
01B3'   7E              ..NEXN: MOV A,M
01B4'   FE02                    CPI $NULL         ;IS IT $NULL?
01B6'   20F3                    JRNZ ..NEXL       ;NOPE, CHAIN ON
01B8'   E5                      PUSH H            ;SAVE CURRENT
01B9'   CD 0000:21              CALL NXTLNK       ;GET NEXT LINK
01BC'   DD7503                  MOV $NLINK(X),L   ;CHAIN OUT
01BF'   DD7404                  MOV $NLINK+1(X),H
01C2'   E1                      POP H             ;GET H BACK
01C3'   F5                      PUSH PSW          ;SAVE CC'S FROM NXTLNK
01C4'   CD 0000:11              CALL FREE         ;FREE IT UP
01C7'   DDE5                    PUSH X            ;RESTORE HL
01C9'   E1                      POP H
01CA'   F1                      POP PSW           ;WASZERO?
01CB'   20E1                    JRNZ ..NEXH
01CD'   F1              ..NEXA: POP PSW           ;GET CURRENT LETTER BACK
01CE'   FE24                    CPI '$' ;WAS $?
01D0'   C8                      RZ                ;YUP-DONE
01D1'   3C                      INR A             ;A GOES TO B, ETC.
01D2'   FE5B                    CPI 'Z'+1
01D4'   20D1                    JRNZ ..NCHR       ;DO IT
01D6'   3E24                    MVI A,'$'
01D8'   18CD                    JMPR ..NCHR

                        ; *LINEP, THE LINE PROCESSOR, FOLLOWS

01DA'                   LINEP:  ;CHECK FOR CNTC
01DA'   CD 0073'                CALL CNTCK
01DD'   CD 0000:13              CALL GETLPT       ;GET LINE POINTER
01E0'   CD 0000:16              CALL ISNL         ;IS END OF COMMAND?
01E3'   C8                      RZ                ;YUP, RETURN
01E4'   FE2E                    CPI '.'           ;COMMENT?
01E6'   2008                    JRNZ LINOO        ;NO
01E8'   CD 0000:2C      LINO2:  CALL SCANNL       ;SKIP TO THE NEXT LINE
01EB'   CD 0000:2A      LINO3:  CALL PUTLPT       ;AND STUFF HL
01EE'   18EA                    JMPR LINEP        ;AND TRY AGAIN
01F0'   CD 0000:1B      LINOO:  CALL KLB          ;ZAP BLANX
01F3'   CD 0000:07              CALL CARTYP       ;CHECK FOR LABEL
                                ZERROR  ER.CMD    ; NO LEADING PUNCTI
01F6'   2004            +       JRNZ ..0001
01F8'   CD 031D'        +       CALL ERRPGM
01FB'   1F              +       .BYTE ER.CMD
01FC'                   +..0001:]
01FC'   3006                    JRNC LINO1        ;ITS NOT
01FE'   CD 0000:2B              CALL SCANLAB      ;SCAN PAST IT
0201'   C8                      RZ                ;RETURN IF ->DELIM
0202'   18E7                    JMPR LINO3        ;AND TRY AGAIN
```

```
0204'  CD 0184'     LINO1:  CALL PRCHEK      ;SEE IF LINE WANTS PRINTED
0207'  CD 0000:1B            CALL KLB        ;ZAP LEAD BLANX
020A'  CD 0000:1E            CALL NAMSET     ;DE->FIRST CHAR,
                                             ;HL->TERMINATOR
020D'  CD 0000:16            CALL ISNL       ;CHECK TERMINATOR
0210'  281C                  JRZ ..CMD       ;IS ;, NL OR NULL
0212'  FE20                  CPI BLANK       ;ISNL LOADS A
0214'  2818                  JRZ ..CMD
0216'  FE2C                  CPI ',          ;ALLOW COMMAS TOO
0218'  2814                  JRZ ..CMD
021A'  FE2E                  CPI EXTDEL      ;SWITCH?
021C'  2810                  JRZ ..CMD
021E'  FE28                  CPI '(          ;ARRAY?
0220'  CC 0000:29            CZ PSCAN
0223'  7E                    MOV A,M         ;PICKUP CHAR
0224'  FE3D                  CPI '=          ;='S?
0226'  2002                  JRNZ ..AASN     ;TRY IT ANYWAY
0228'  365F          ..OK:   MVI M,AASN      ;STUFF A AASN
022A'  EB            ..AASN: XCHG            ;RESTORE H
022B'  C3 0000:0F            JMP EVALARG
022E'  AF            ..CMD:  XRA A           ;TRY SYS NAMES
022F'  CD 0000:1D            CALL NAMADR     ;IFZSET, GOOD
0232'  2804                  JRZ DOCMD
0234'  CD 0316'              CALL ERRPGX
0237'  23                    .BYTE ER.NOTF

                     ;ENTRY TO PROCESS ANYTHING GIVEN NAMBLOK ADDR

0238'  DDE5          DOCMD:  PUSH X          ;SAVE X
023A'  FDE5                  PUSH Y          ;AND Y
023C'  3A 6260               LDA CSFLAG      ;ONLY STORE SP
023F'  B7                    ORA A           ;IF CSFLAG OFF
0240'  2012                  JRNZ ..NOCS     ;DON'T DO IT
0242'  ED73 625C             SSPD SAVESP
0246'  DD2A 655B             LIXD OPRSP      ;SAVE OP STACK PTR
024A'  DD22 653D             SIXD SOPRSP     ;FOR ERROR RECOVERY
024E'  3A 655D               LDA OPRSZ       ;DITTO FOR SIZE
0251'  32 653F               STA SOPRSZ      ;OF OPR STACK
0254'  DD21 027B'    ..NOCS: LXI X,RETONE    ;PUSH RET ADDR
0258'  DDE5                  PUSH X
025A'  1A                    LDAX D          ;GET FWA
025B'  FE18                  CPI $CMDADR     ;IS COMMAND?
025D'  203A                  JRNZ DOMACRO    ;NOPE, TRY MACROS
025F'  EB                    XCHG            ;DE->TERMINATOR,
0260'  60                    MOV H,B         ;BC->POINTER TO CODE
0261'  69                    MOV L,C
0262'  CD 0000:22            CALL NXTPTR     ;GO TO IT
0265'  EB                    XCHG            ;DE->CODE, HL->TERM
0266'  D5                    PUSH D          ;DE IS DISPATCH ADDR!
0267'  C9                    RET             ;GOTO IT, CMDREN STILL TOS

                     ;       CMDREN--GENERAL END OF COMMAND STUFF
                     ;DETAILS:
                     ;       DEFAULT RETURN IS TO RETONE
                     ;         WHICH RETURNS INTEGER 1
```

```
                        ;       RETNONE RETURNS WHATEVER A&DE HAVE
                        ;       CMDSKO  SETS UP R#TO REDO CURRENT LINE
                        ;       CMDSKP  SETS UP TO SKIP REST OF LINE


0268'   CD 0000:13      CMDSKO: CALL GETLPT     ;GET OLD LINE PTR
026B'   2B                      DCX H           ;BACKUP ONE IF NOT
026C'   7E                      MOV A,M         ;FIRST LINE
026D'   B7                      ORA A           ;IE, HL->NL
026E'   200A                    JRNZ CMDRES     ;IF HL->NULL,
0270'   23                      INX H           ;FIX IT
0271'   1807                    JMPR CMDRES
0273'   CD 0000:2C      CMDSKP: CALL SCANNL     ;ENTER HERE TO SKIP
0276'   B7                      ORA A           ;DCX IF NON-NULL
0277'   2801                    JRZ CMDRES      ;...
0279'   2B                      DCX H           ;REST OF LINE
027A'   C1              CMDRES: POP B           ;ENTER HERE TO STRIP
                                                ;OFF RET TO RETONE
027B'   3E04            RETONE: MVI A,$IVAL     ;INTEGER 1 IN DE
027D'   11 0001                 LXI D,1         ;AND FALL THRU
0280'   1801                    JMPR CMDREN
0282'   C1              RETNONE:        POP B   ;DUMP RET TO RETONE

0283'   F5              CMDREN: PUSH PSW        ;SAVE A
0284'   CD 0000:2A              CALL PUTLPT     ;STUFF HL INTO $LINPTR
0287'   CD 0162'                CALL STUFFCHK   ;ANYBODY PENDING?
028A'   CD 0000:0A              CALL CORCHK
                                CERROR ER.CORF[
028D'   3004            +       JRNC ..0002
028F'   CD 031D'        +       CALL ERRPGM
0292'   1B              +       .BYTE ER.CORF
0293'                   +..0002:]
0293'   F1                      POP PSW         ;RESTORE A
0294'   FDE1                    POP Y           ;STILL ON STACK
0296'   DDE1                    POP X           ;RESTORE X
0298'   C9                      RET

                        ;   *DOMACRO--MACRO EXECUTION STARTUP*
                        ;       NOT A GOOD ENTRY POINT (STUFF ON STACK)

0299'   CD 0000:2A      DOMACRO:CALL PUTLPT     ;SAVE CURRENT POINTER
029C'   E5                      PUSH H          ;SAVE H FOR ARRAYZ
029D'   62                      MOV H,D         ;POINT HL AT NAMEBLOCK
029E'   6B                      MOV L,E         ;LEAVE IN DE FOR ARRAYS
029F'   CD 0000:23              CALL NXTVAL     ;GET STRING FWA
02A2'   7E                      MOV A,M         ;GET TYPE
02A3'   FE08                    CPI $STRADR     ;IS STRING?
02A5'   204E                    JRNZ DOARA
02A7'   E1                      POP H           ;GET HL BACK
02A8'   DDE5                    PUSH X          ;SAVE X
02AA'   7E                      MOV A,M         ;CHECK FER .B
02AB'   FE2E                    CPI EXTDEL      ;IS IT?
02AD'   201A                    JRNZ ..FORE     ;NOPE, REGULAR CALL
02AF'   23                      INX H           ;SKIP EXTDEL
02B0'   23                      INX H           ;SKIP B
```

```
02B1'   CD 0000:1C          CALL MAKMIB      ;BUILD MIB
02B4'   ED4B 62E8           LBCD DOLPLH      ;GET DOL LISTHEAD
02B8'   DD710F              MOV $BGPTR(X),C  ;LINK IN
02BB'   DD7010              MOV $BGPTR+1(X),B
02BE'   DDCB02D6            SET $DOLOOP,$FLAGS(X)   ;SAY IS DOLOOP'D
02C2'   DD22 62E8           SIXD DOLPLH      ;LINK US IN
02C6'   DDE1                POP X            ;RESTORE X
02C8'   C9                  RET              ;RETURN SUCCESS
02C9'   CD 0000:1C  ..FORE: CALL    MAKMIB   ; CALL MIB MAKE ROUTINE
02CC'   CD 0000:2A          CALL PUTLPT      ;SAVE EOL FOR RET TIME
02CF'   ED4B 625E           LBCD MACTOP      ;GET CURRENT MACRO
02D3'   DD710D              MOV $CALLER(X),C        ;POINT TO CALLER
02D6'   DD700E              MOV $CALLER+1(X),B
02D9'   DD22 625E           SIXD MACTOP      ;MACTOP IS US NOW
02DD'   DDE1                POP X
02DF'   F5                  PUSH PSW         ;EXTRA FOR GOSUB COMPATS
02E0'           DOMORE:                      ;ENTRY FROM GOSUB
02E0'   CD 01DA'            CALL LINEP       ;PROCESS LINE
02E3'   CD 0000:17          CALL ISTERM      ;CHECK TERMY
02E6'   B7                  ORA A            ;CHECK FER NULL
02E7'   2007                JRNZ ..CZC       ;NOPE
02E9'   D1                  POP D            ;GOSUB RETURN(?)
02EA'   CD 0000:2E          CALL SWITCH      ;SWITCHEROO
02ED'   C3 0282'            JMP RETNONE      ;AND PASS A&DE BACK
02F0'   CD 0304'    ..CZC:  CALL CZCHEK      ;CHECK DDT&CNTRLS
02F3'   18EB                JMPR DOMORE      ;CONTINUE

                    ; ARVAL WANTS DE->NAMEBLOCK, HL->AFTER PAREN

02F5'   E1          DOARA:  POP H            ;GET PTR TO ARGS BACK
02F6'   FE10                CPI $AADR        ;IS ARRAY
                            NZERROR ER.MACI
02F8'   2804        +       JRZ ..0003
02FA'   CD 031D'    +       CALL ERRPGM
02FD'   22          +       .BYTE ER.MAC
02FE'               +..0003:]
02FE'   CD 0000:06          CALL ARVAL       ;DO IT
0301'   C3 0282'            JMP RETNONE      ;LEAVE WITH A&DE SETUP

                    ;*CZ CHEK--CHECKS FOR CONTROL Z AND DEBUG

0304'   3A 6260    CZCHEK: LDA CSFLAG
0307'   B7                 ORA A
0308'   2006               JRNZ ..NOCZ      ;DON'T CHECK DEBUG IF
030A'   3A 65CE            LDA DDTON        ;IN CNTRLZ ALREADY
030D'   B7                 ORA A            ;IS DEBUG ON? (CNTRLD)
030E'   2070               JRNZ ZSIN
0310'   CD 0073'    ..NOCZ: CALL CNTCK      ;CHECK'EM
0313'   206B               JRNZ ZSIN
0315'   C9                 RET

                    ;*ERROR--ERROR MESSAGE HANDLER & CNTLS HANDLER*
0316'   E1          ERRPGX: POP H            ;POP CALL RET

0317'   DDE5                PUSH X
```

```
0319'   FDE5                   PUSH Y
031B'   1805                   JMPR ERRFOO      ;KLUDGE FOR LINEP ERRMESS
031D'   E3            ERRPGM:  XTHL
031E'   ED7B 625C              LSPD SAVESP      ;FIX STACK UP
0322'   ED4B 653D    ERRFOO:  LBCD SOPRSP      ;RESTORE OP STACK
0326'   ED43 655B              SBCD OPRSP       ;POINTER
032A'   3A 653F                LDA SOPRSZ       ;DITTO FOR STACK
032D'   32 655D                STA OPRSZ        ;BYTE
0330'   7E                     MOV A,M          ;GET ERROR NUMBER
0331'   F5                     PUSH PSW         ;SAVE ERROR NUMBER
0332'   32 672A                STA ONEBUF+1     ;STORE ERROR NUMBER
0335'   3A 6729                LDA ONEBUF       ;GET ONERROR FLAG
0338'   B7                     ORA A            ;IS ZILCH?
0339'   2803                   JRZ ..PRNT       ;YUP-NORMAL
033B'   F1                     POP PSW          ;NOPE, ONERROR TIME
033C'   1817                   JMPR ..CHK2
033E'   CD 0197'      ..PRNT:  CALL PRLINE      ;PRINT LINE IN ERROR
0341'   CD 0000:28              CALL PRINTFF
0344'   3F3F4552524F            .ASCIZ /??ERROR #/
034E'   F1                     POP PSW
034F'   CD 0000:25              CALL PDECA       ;A HAS ERROR MESSAGE
0352'   CD 0000:1F              CALL NLPNT
0355'   ED4B 625E     ..CHK2:  LBCD MACTOP      ;CHECK IF KB LEVEL
0359'   0A                     LDAX B
035A'   FE26                   CPI $KEYBLOCK    ;IS IT?
035C'   C3 00A2'               JMP ABORT        ;TEMPORARY TILL BELOW WORKS
035F'   CA 00A2'               JZ ABORT         ;YEAH, ABORT
0362'   3A 6260                LDA CSFLAG       ;IS SECOND TIME THRU?
0365'   B7                     ORA A            ;I.E. IS CSFLAG SET?
0366'   200C                   JRNZ ..TRY       ;YUP, DON'T CHANGE PTR
0368'   CD 0000:13    ..GLPT:  CALL GETLPT      ;GET CURRENT LINE
036B'   22 672B                SHLD ONEBUF+2    ;STORE LINE POINTER
036E'   CD 0000:2C              CALL SCANNL      ;TO NEXT ONE
0371'   CD 0000:2A              CALL PUTLPT      ;SAVE POINTER
0374'   ED7B 625C     ..TRY:   LSPD SAVESP      ;RESTORE STACK
0378'   FDE1                   POP Y
037A'   DDE1                   POP X            ;RET STILL ON STACK
037C'   21 01DA'               LXI H,LINEP      ;FAKE NEW RET
037F'   E5                     PUSH H           ;ONTO STACK
0380'   CD 008D'      ZSIN:    CALL ZAPCZ       ;CLEAN UP MIBS IF NECESSARY
0383'   21 668D                LXI H,CSBLOK     ;BUILD MIB
0386'   CD 0000:18              CALL KBMIB
0389'   3628                   MVI M,$CSBLOK    ;TAG IT
038B'   AF             ..SWIN:  XRA A            ;NULL ON/OFF'S
038C'   32 65D9                STA CNTRLO       ;LIKE THIS
038F'   32 65E4                STA CNTRLZ       ;ZAP CNTRLZ
0392'   3A 6729                LDA ONEBUF       ;ONERROR ON?
0395'   B7                     ORA A
0396'   202B                   JRNZ ..ONDO      ;WELL, DO IT
0398'   3E23                   MVI A,'#'        ;PUT OUT HASH
039A'   32 6260                STA CSFLAG       ;SET CSFLAG#0
039D'   CD 0000:24              CALL OUTCH
03A0'   21 66A8                LXI H,CSBLOK+$MIBEND    ;BUFFER ADDR
03A3'   3600                   MVI M,0          ;NULL COUNT
03A5'   CD 0073'       ..SWT:   CALL CNTCK       ;CHECK ^C
```

```
03A8'  CD 0000:15          CALL INPTTY      ;GET A LINE
03AB'  20F8                JRNZ ..SWT       ;WAIT FER IT
03AD'  23                  INX H            ;POINT PAST COUNT
03AE'  7E                  MOV A,M          ;NULL LINE?
03AF'  B7                  ORA A
03B0'  282C                JRZ ..BACK       ;YEP- FAKE RETURN
03B2'  CD 01DA'     ..SGO: CALL LINEP       ;DO THE DEED
03B5'  CD 0000:17          CALL ISTERM      ;CHECK TERMY
03B8'  B7                  ORA A
03B9'  20F7                JRNZ ..SGO       ;STUFF LEFT
03BB'  21 66A9             LXI H,CSBLOK+$MIBEND+1  ;SET LINEPTR
03BE'  CD 0000:2A          CALL PUTLPT      ;LIKE THIS
03C1'  18C8                JMPR ..SWIN      ;MORE TO GO
03C3'  32 6260      ..ONDO: STA CSFLAG       ;SET CSFLAG
03C6'  AF                  XRA A            ;CLEAR ONEBUF
03C7'  32 6729             STA ONEBUF
03CA'  11 66A9             LXI D,CSBLOK+$MIBEND+1
03CD'  21 672D             LXI H,ONEBUF+4
03D0'  01 0008             LXI B,8
03D3'  EDB0                LDIR
03D5'  CD 01DA'     ..OOOO: CALL LINEP      ;EXECUTE THE LINE
03D8'  CD 0000:17          CALL ISTERM      ;DONE?
03DB'  B7                  ORA A
03DC'  20F7                JRNZ ..OOOO      ;MORE
03DE'  AF           ..BACK: XRA A            ;CLEAR CSFLAG
03DF'  32 65E4             STA CNTRLZ       ;AND CNTRLZ
03E2'  32 6260             STA CSFLAG       ;SO CAN ^Z AGAIN
03E5'  C3 0000:2E          JMP SWITCH       ;BACK TO INTERRUPTEE


                    ; *RESTART--SYSTEM RESET CODE*
03E8'               .MAIN.:
03E8'  F3           RESTART:        DI       ;DISABLE INTERRUPTS
03E9'  31 60C8              LXI     SP,STACK
03EC'  AF                   XRA A            ;AND ZAP INMOD
03ED'  D3F0                 OUT     0F0H     ; NO WIERD MEMORY PLEASE
03EF'  D3F1                 OUT     0F1H     ; KILL DISC
03F1'  D30E                 OUT INMOD        ;...
03F3'  D320                 OUT     NORML
03F5'  D32A                 OUT     2AH
03F7'  D308                 OUT     8
03F9'  D3F9                 OUT     0F9H     ; 32-64 INTERNAL
03FB'  3E0F                 MVI     A,0FH
03FD'  D3FA                 OUT     0FAH
03FF'  AF                   XRA     A
0400'  D3FB                 OUT     0FBH     ; 32-64 WRITE ENABLED
0402'  3EFF                 MVI     A,0FFH
0404'  D324                 OUT     LEDS     ; CLEAR LEDS
0406'  3E11                 MVI     A,11H    ; 0-20K EXTERNAL
0408'  D3F8                 OUT     0F8H
040A'  3E03                 MVI     A,3
040C'  D3FF                 OUT     0FFH
                    ;HYPE MUSIC AMPLITUDE
040E'  3E3F                 MVI A,03FH
0410'  D322                 OUT     BSAUD
```

```
0412'  CD 0468'            CALL    INUART
                    ; INITIALIZE INTERRUPTS
0415'  21 0016'            LXI     H,ZGITAB
0418'  7C                  MOV     A,H
0419'  ED47                STAI
041B'  7D                  MOV     A,L
041C'  D30D                OUT     INFBK
041E'  ED5E                IM2
0420'  3EC8                MVI     A,200   ; PROGRAM VERTICAL INTERRUPT
0422'  D30F                OUT     INLIN
0424'  3E08                MVI A,8
0426'  D30E                OUT INMOD       ;ENABLE SCREEN INTS,
                                           ;AND INT EVERY M1
0428'  21 60CA             LXI H,CLEARS
042B'  01 1F35             LXI B,RAMEND-CLEARS     ;LENGTH OF USER
                                                   ;STORAGE
042E'  CD 0000:08          CALL CLEARIT    ;PROPAGATE ZEROES
0431'  CD 0480'            CALL    INFART
0434'  CD 0000:0B          CALL CORINI     ;STRUCTURE USER MEMORY
0437'  CD 0000:14          CALL    INIVDR  ; SET UP ONSCREEN TYPING

                    ;       NOW BUILD INTEGER A-Z ENTRIES/HASH TABLE
043A'  3E41                MVI A,'A'       ;START WITH A
043C'  21 62EC             LXI H,FSTINT    ;FIRST INTEGER ADDRESS

043F'  11 0000:09          LXI D,CMDTAB    ;TABLE OF COMMAND ADDRS
0442'  F5           ..REST: PUSH PSW
0443'  360E                MVI M,$FADR     ;SET TO IVAL
0445'  23                  INX H
0446'  3601                MVI M,1         ;SIZE IS 1 BLOCK
0448'  23                  INX H
0449'  3680                MVI M,$NDEL     ;SET BYTE 2 TO $NDEL
044B'  01 0007             LXI B,$NASCII-2 ;PUT IN NAME
044E'  09                  DAD B
044F'  77                  MOV M,A         ;LIKE THIS
0450'  01 0005             LXI B,14-$NASCII        ;->TO CMD PLACE
0453'  09                  DAD B

0454'  EB                  XCHG
0455'  01 0002             LXI B,2
0458'  EDB0                LDIR            ;COPY CMDTAB ENTRY
045A'  EB                  XCHG            ;HL<->DE
045B'  FE24                CPI '$'         ;DID $?
045D'  2845                JRZ REBDON      ;DONE
045F'  3C                  INR A           ;A CHANGES TO B, ETC.
0460'  FE5B                CPI 'Z'+1       ;DONE?
0462'  20DE                JRNZ ..REST
0464'  3E24                MVI A,'$'
0466'  18DA                JMPR ..REST     ;DO $
                    ; ROUTINE TO INITIALIZE ADDON UART
0468'               INUART:
0468'  3EAE                MVI     A,0AEH
046A'  D32B                OUT     AUARTS
046C'  3E40                MVI     A,40H
046E'  D32B                OUT     AUARTS
```

.MAIN. - ZGRASS--THIS IS IT, THE VERY BEST PART OF THE TRIP
LINEP--THE LINE PROCESSOR

```
0470'   3ECE                    MVI     A,0CEH
0472'   D32B                    OUT     AUARTS
0474'   3E27                    MVI     A,27H
0476'   D32B                    OUT     AUARTS
0478'   DB2A                    IN      AUARTD
047A'   DB2A                    IN      AUARTD
                        ; SET BAUD RATE AT 2400
047C'   3E38                    MVI     A,00111000B
047E'   D323                    OUT     BAUDSEL
0480'   21 64B3          INFART: LXI     H,TAPBUF
0483'   22 64AF                    SHLD    TAPCON
0486'   22 64B1                    SHLD    TAPPRO
0489'   21 0489'         ..SELF: LXI     H,..SELF
048C'   3E16                    MVI     A,16H
048E'   CB74                    BIT     6,H
0490'   2802                    JRZ     ..OLD
0492'   3E14                    MVI     A,14H
0494'   D321             ..OLD: OUT     VCTR
0496'   C9                      RET
                        ; ROUTINE TO INITIALIZE DEVICE VARIABLES!
0497'   11 6579          RGBINI: LXI     D,DEVVAR        ; DE=INITIAL DEVICE TABL
                        E
049A'   21 004C'                LXI     H,INIDEV        ; HL=INITIAL VALUES
049D'   01 001A                 LXI     B,ENDDEV-INIDEV
04A0'   EDB0                    LDIR
04A2'   FB                      EI
04A3'   C9                      RET
04A4'   CD 0497'         REBDON: CALL RGBINI
04A7'   CD 0000:28              CALL PRINTFF
04AA'   0C                      .BYTE   CNTRL
04AB'   475241464958            .ASCIZ  /GRAFIX
04B1'   0D0A00          /
04B4'   C3 00A2'                JMP ABORT       ;AND DO IT
                        .END
```

.MAIN. - ZGRASS--THIS IS IT, THE VERY BEST PART OF THE TRIP
+++++ SYMBOL TABLE +++++

```
AASN    005F          ABORT   00A2'   I     ALLOC   0000:04 X     ARG     0000:05 X
ARGLIN  0197'   I     ARGSTK  60CC          ARVAL   0000:06 X     AUARTD  002A
AUARTS  002B          BACKGR  65CC          BAUDSE  0023          BLANK   0020
BOTRAM  6000          BOTTOM  64A9          BSAUD   0022          CARTYP  0000:07 X
CHARSL  65DD          CLEARI  0000:08 X     CLEARS  60CA          CMDREN  0283'   I
CMDRES  027A'   I     CMDSKO  0268'   I     CMDSKP  0273'   I     CMDTAB  0000:09 X
CNTCK   0073'   I     CNTRL   000C          CNTRLC  65CD          CNTRLO  65D9
CNTRLU  0015          CNTRLZ  65E4          CORCHK  0000:0A X     CORINI  0000:0B X
CPLARE  611C          CPLSIZ  0140          CR      000D          CSBLOK  668D
CSFLAG  6260          CURCX   6814          CURCY   6816          CURREN  64A5
CZCHEK  0304'   I     C.CO    0011          C.C1    0012          C.CX    000B
C.CY    000D          C.DP    0013          C.ST    0002          C.X     0003
C.XF    000F          C.XS    0007          C.Y     0005          C.YF    0010
C.YS    0009          DDTON   65CE          DELMIB  0000:0C X     DEVBL   6589
DEVCL0  6579          DEVCL1  657B          DEVCL2  657D          DEVCL3  657F
DEVCL4  6581          DEVCL5  6583          DEVCL6  6585          DEVCL7  6587
DEVFB   65CB          DEVHCB  658F          DEVMO   658B          DEVNM   65B7
DEVNT   658D          DEVTNA  65BB          DEVTNB  65BF          DEVTNC  65C3
DEVVA   65BD          DEVVAR  6579          DEVVB   65C1          DEVVBL  6591
DEVVC   65C5          DEVVD   65C9          DEVVN   65B9          DEVVS   65C7
DEVXCD  65B3          DEVYCD  65B5          DNULLS  01A5'         DOARA   02F5'
DOCHK   0000:0D X     DOCMD   0238'   I     DOLPLH  62E8          DOLPPT  62EA
DOMACR  0299'         DOMORE  02E0'   I     DUMBST  6577          EDBCNT  64AD
EDCCNT  64A7          EDLONG  6812          EDMODE  681C          EDNAME  64A1
EDNCX   680C          EDNCY   680E          EDNEWS  64A5          EDOCX   6808
EDOCY   680A          EDPN    6806          EDPO    6804          EDPTRC  64AB
EDPTRL  64A9          EDSTR   681A          ENDDEV  0066'         ERABIT  0002
ERRFOO  0322'         ERRPGM  031D'   I     ERRPGX  0316'         ER.ARA  002F
ER.ARG  0034          ER.ASN  0015          ER.BOX  001A          ER.CHN  0002
ER.CMD  001F          ER.CNV  0016          ER.COR  001B          ER.CTL  0036
ER.DEL  0026          ER.DIM  0030          ER.DIV  0018          ER.DP   0037
ER.DSK  0019          ER.EDT  0035          ER.FMT  0038          ER.FNF  001C
ER.FOR  0028          ER.IMP  0003          ER.LAB  0025          ER.MAC  0022
ER.NAE  002D          ER.NAM  0029          ER.NEG  003A          ER.NOT  0023
ER.NUL  0039          ER.NUM  002B          ER.NXT  001D          ER.OFL  0017
ER.OPN  0014          ER.OVE  001E          ER.PAR  002A          ER.REN  002C
ER.RET  0024          ER.SEP  0021          ER.SNP  0031          ER.SPC  002E
ER.STK  0004          ER.SW   0032          ER.TER  0020          ER.UFL  0033
ER.UNF  0027          EVAL    0000:0E X     EVALAR  0000:0F X     EXTDEL  002E
E.HVAL  0002          E.LVAL  0001          E.SIZ   0005          E.TYP   0000
E.VAL   0001          FCNTH   65D2          FCNTI   65D3          FCNTJ   65D4
FCNTK   65D5          FCNTL   65D6          FCNTV   65E0          FCNTY   65E3
FINDHA  0000:10 X     FIRST   64A3          FLAGS   65CB          FOREGR  65D0
FRAGSI  0400          FREE    0000:11 X     FREEAL  0000:12 X     FREELS  65E5
FSTDOL  648C          FSTINT  62EC          FWDPTR  65E7          GETLPT  0000:13 X
HCAREA  6593          HORCB   0009          INCRO   65E9          INFART  0480'
INFBK   000D          INIDEV  004C'   I     INIVDR  0000:14 X     INLIN   000F
INMOD   000E          INPTTY  0000:15 X     INTTAB  0012'         INUART  0468'
ISNL    0000:16 X     ISTERM  0000:17 X     JUNK    6542          KBLOCK  65F1
KBMIB   0000:18 X     KEYFLG  67FF          KEYPTK  6533          KEYTRK  67F7
KILIST  007E'         KILMIB  0000:19 X     KISNL   0000:1A X     KLB     0000:1B X
LEDS    0024          LF      000A          LINOO   01F0'         LINO1   0204'
LINO2   01E8'         LINO3   01EB'         LINEP   01DA'   I     LISTON  65E2
MACSTU  6536          MACTOP  625E          MAGIC   000C          MAKMIB  0000:1C X
MAXFRG  0040          MNMX    65EB          MPBAV   0016          MPMO    0010
```

.MAIN. - ZGRASS--THIS IS IT, THE VERY BEST PART OF THE TRIP
+++++ SYMBOL TABLE +++++

MPNCV	0015	MPNV	0017	MPTNA	0011	MPTNB	0012
MPTNC	0013	MPVIB	0014	NAMADR	0000:1D X	NAMSET	0000:1E X
NBLKB	0000	NBLKM	0001	NEWBOT	6810	NL	000A
NLPNT	0000:1F X	NMIROU	006A'	NORML	0020	NSADDR	6802
NUMBUF	64A3	NXTABC	0000:20 X	NXTLNK	0000:21 X	NXTPTR	0000:22 X
NXTVAL	0000:23 X	OLDCHR	64A0	OLDCUR	649F	OLDKEY	649D
OLDXY	65EF	ONEBUF	6729	OPRL	0014	OPRSP	655B
OPRSTK	6547	OPRSZ	655D	OUTCH	0000:24 X	OUTOFF	62E7
PCNT	655E	PDECA	0000:25 X	PIXVAL	65ED	POINTE	64A7
PONOFF	65DA	PRCHEK	0184'	PRINT	0000:26 X	PRINTA	0000:27 X
PRINTF	0000:28 X	PRINTR	6540	PRLINE	0197' I	PSCAN	0000:29 X
PUTLPT	0000:2A X	RAMEND	7FFF	RAMSTR	6900	RANSHT	655F
REBDON	04A4'	RESDDT	0070'	RESTAR	03E8' I	RETNON	0282' I
RETONE	027B' I	RGBINI	0497' I	RMDTMP	6538	RUBACK	0005
RUBOUT	007F	SAVESP	625C	SCANLA	0000:2B X	SCANNL	0000:2C X
SCLEAR	0000:2D X	SCNEED	0004	SCRWIN	67CD	SOPRSP	653D
SOPRSZ	653F	STACK	60C8	STAKTO	6000	STRSIZ	6800
STUFFC	0162'	SUBSTU	6534	SWITCH	0000:2E X	TAB	0009
TAPBUF	64B3	TAPCON	64AF	TAPPRO	64B1	TBFEND	6533
TEMPHD	60CA	TEMPS	62E5	TMPARG	67AD	TOP	6818
TTYBEG	6261	TTYEND	62E1	TTYINT	62E3	TTYPTR	62E1
TXTWIN	67E2	UARTD	00E0	UARTFL	649C	UARTIN	0000:2F X
UARTS	00E1	USREND	65F1	V3PTR	6573	VBLANK	000A
VCTR	0021	VDCHAR	649E	VDNLF	65CF	VIPLH	6545
VOICEO	6563	WRMODE	65EE	XPAND	0019	ZAPCZ	008D'
ZBEGIN	0004'	ZGIM2	0001	ZGINT	0000:30 X	ZGITAB	0016'
ZGRASS	0102'	ZGREND	681D	ZGRIN	0158'	ZLINEP	0139'
ZLOOP	0142'	ZSIN	0380'	$AADR	0010	$ADDRF	0007
$ADDRI	0005	$ADDRS	0009	$ANY	001A	$ANYNA	FFFC
$ANYVA	FFFE	$ARGPT	0011	$BGPTR	000F	$BNDL	0007
$CALLE	000D	$CMDAD	0018	$CPLBL	002A	$CSBLD	0028
$DATAP	0007	$DOLDE	0001	$DOLDO	0002	$DVAL	0000
$END	001C	$FADR	000E	$FLAGS	0002	$FORBL	0024
$FORPT	000B	$FVAL	0006	$GOSUB	001A	$IADR	000C
$INPBU	0018	$INPPT	0016	$IVAL	0004	$KEYBL	0026
$LENGT	0001	$LINPT	0005	$LOCPT	0009	$MIBBL	0022
$MIBEN	001B	$NAMAD	000A	$NAME	000A	$NASCI	0009
$NDEL	0080	$NLINK	0003	$NULL	0002	$NVALU	0005
$REPEA	001E	$RVSTU	0013	$SAME	0020	$SASCI	000A
$SLEN	0006	$STRAD	0008	$STRPT	0003	$TAF	0000
$TYPE	0000	$USE	0005	.BLNK.	0000:03 X	.DATA.	0000* X
.MAIN.	03E8' I	.PROG.	04B7' X				