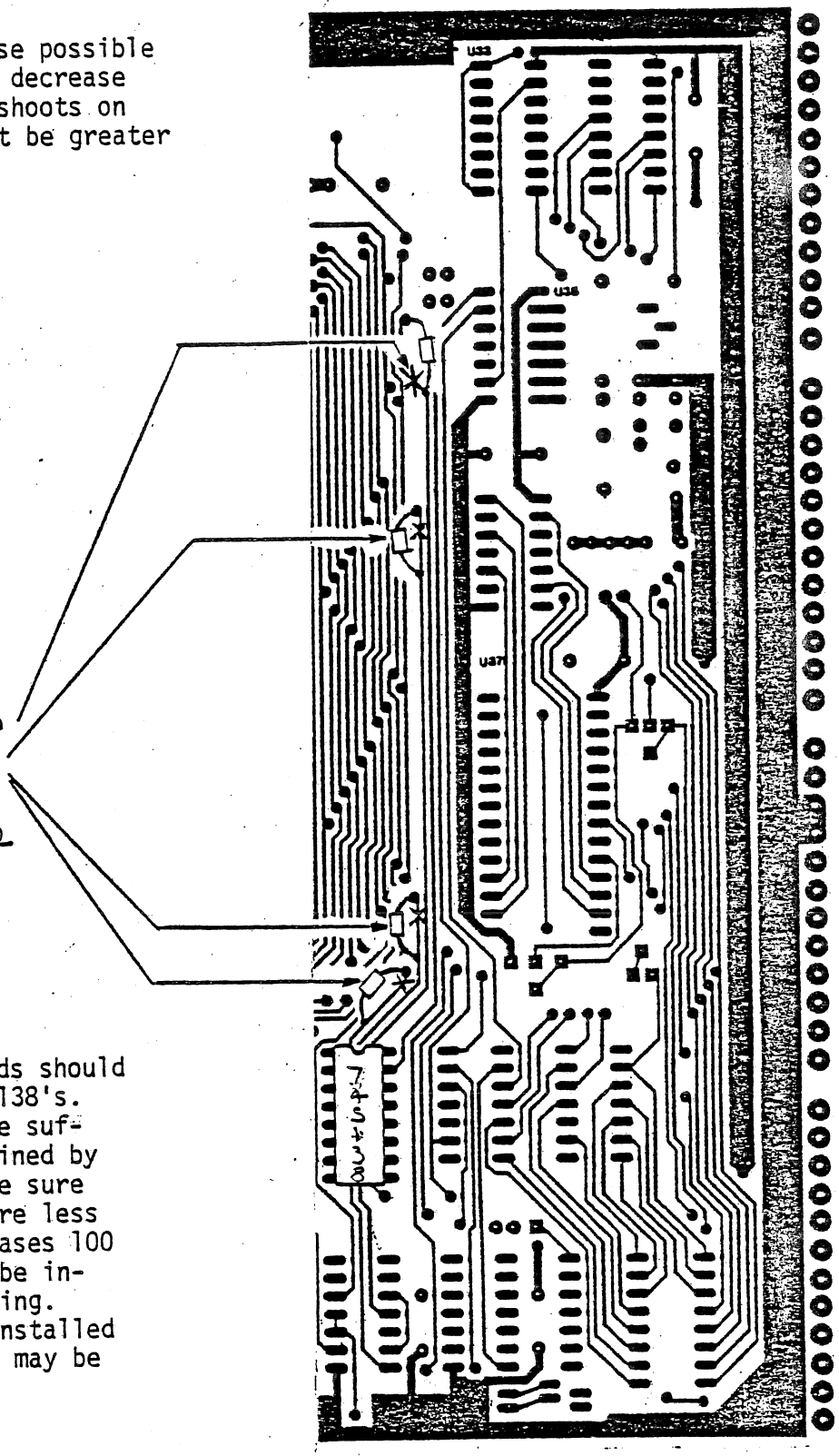
OHIO SCIENTIFIC TECH NEWSLETTER #19

September 7, 1979

530 BOARD PRODUCT UPDATE

PURPOSE OF CHANGE: To decrease possible noise on the 530 Board and to decrease possible negative going undershoots on the $\overline{RAS}$ lines. These must not be greater than -1.0 volt.



CHANGE: All 530 Memory Boards should use 74LS138's instead of 74S138's. In some cases this may not be sufficient. This can be determined by scoping the $\overline{RAS}$ lines to make sure negative going undershoots are less than -1.0 volts. In these cases 100 ohm series resistors should be installed as shown in the drawing. When 100 ohm resistors are installed either 74S138's or 74LS138's may be used.

OS-65U STRING VARIABLES

THIS WEEK WE SHALL TAKE A LOOK AT GARBAGE COLLECTION OF STRING DATA IN 9 DIGIT BASIC. STRING DATA IS STORED IN TWO AREAS OF MEMORY. A STRING VARIABLE EQUATED TO DATA IN A READ STATEMENT OR TO A QUOTED STRING HAS A DESCRIPTOR THAT POINTS AT THE STRING DATA WITHIN THE PROGRAM. THE STRING VARIABLE DIRECTLY OR INDIRECTLY EQUATED TO AN INPUT STATEMENT WILL HAVE A DESCRIPTOR POINTING INTO STRING SPACE. IN OTHER WORDS, ANY STRING VARIABLE THAT IS EQUATED TO STRING DATA NOT IN A DATA STATEMENT OR TO STRING DATA IN QUOTES, WILL HAVE A DESCRIPTOR THAT POINTS INTO STRING SPACE. IT IS THE CONSTRUCTION OF STRINGS IN STRING SPACE THAT CAUSES "GARBAGE STRINGS". A "GARBAGE STRING" WILL BE CREATED IF:

1) A STRING VARIABLE IS EQUATED MORE THAN ONE TIME TO A STRING IN STRING SPACE.

2) A STRING VARIABLE IS EQUATED TO A MID$, LEFT$, OR RIGHT$ EXPRESSION.

3) A STRING VARIABLE IS EQUATED TO A STRING CONCATENATION, I.E. A$ = A$ + B$ OR A$ = A$ + "TEST", ETC.

AN EXAMPLE MAY CLARIFY THE PROCESS.

A$ HAS BEEN PREVIOUSLY EQUATED IN AN INPUT STATEMENT. THE STRING DESCRIPTOR FOR A$ POINTS AT THE ACTUAL STRING DATA STORED IN STRING SPACE (SEE DIAGRAM 1 TECH NEWSLETTER #18). NOW IF A$ IS EQUATED TO AN EXPRESSION LIKE A$ = "NEW" + A$, THE FOLLOWING WOULD OCCUR. FIRST THE STRING EXPRESSION WOULD BE EVALUATED AND A TEMPORARY DESCRIPTOR FOR "NEW" WOULD BE CONSTRUCTED. NEXT THE STRING DATA OF A$ WOULD BE APPENDED TO THE TEMPORARY STRING CONTAINING "NEW". FINALLY THE NEW STRING WOULD BE COPIED INTO STRING SPACE. A$'S DESCRIPTOR WOULD BE UPDATED TO POINT AT THE NEW STRING. THE FREE MEMORY POINTER WOULD BE ADJUSTED TO POINT JUST BELOW THE NEW STRING AND THE TEMPORARY STRINGS WOULD BE "THROWN AWAY".

THOUGH THIS IS NOT THE EXACT SEQUENCE OF EVENTS, IT IS CLOSE ENOUGH FOR OUR DISCUSSION. AT THIS POINT WE HAVE A NEW STRING POINTED TO BY A$'S DESCRIPTOR, BUT WHAT ABOUT THE OLD STRING. AS FAR AS BASIC IS CONCERNED THAT STRING IS "GARBAGE". SINCE BASIC NO LONGER HAS A POINTER TO THE OLD STRING, IT DOES NOT EXIST. IT IS THIS PROCESS OF "REEQUATING" STRINGS (AS WELL AS THE MID$, RIGHT$, AND LEFT$ FUNCTIONS) THAT PRODUCES "GARBAGE STRING DATA".

DIAGRAM 1A SHOWS THE STRING SPACE BEFORE THE STRING CONCATENATION. DIAGRAM 1B SHOWS THE NEW A$ AFTER THE CONCATENATION. NOTE THAT THE ORIGINAL A$ DATA IS LOST SINCE BASIC NO LONGER HAS A POINTER TO IT. (FRETOP), I.E. THE ADDRESS POINTED TO BY THE CONTENTS OF FRETOP, POINTS TO THE FIRST BYTE BELOW THE LAST BYTE OF A$. NEW STRINGS TO BE PLACED IN STRING SPACE WILL ALWAYS BEGIN AT THE CURRENT LOCATION POINTED TO BY (FRETOP). AS STRINGS ARE ADDED TO STRING SPACE, STRING SPACE MOVES DOWN TOWARD THE TOP OF THE ARRAY DESCRIPTORS (STREND). IN OTHER WORDS THE ADDRESS POINTED TO BY (FRETOP) MOVES DOWN IN MEMORY. WHEN (FRETOP) POINTS TO AN ADDRESS THAT IS WITHIN 255 BYTES OF THE END OF THE ARRAY DESCRIPTORS, THE GARBAGE COLLECTION ROUTINE IS CALLED. GARBAGE COLLECTION "RUNS" THROUGH ALL STRING VARIABLE DESCRIPTORS AND MOVES THE ACTUAL STRING DATA UP TO THE TOP OF MEMORY. THIS IS DONE IN SUCH A MANNER THAT ALL ACTIVE STRING DATA (A DESCRIPTOR POINTS TO IT) IS CONTIGUOUS IN MEMORY. GARBAGE COLLECTION ALSO UPDATES THE POINTERS IN ALL STRING DESCRIPTORS SO THAT THEY POINT TO THE STRING DATA AT IT'S NEW LOCATION.

THE GARBAGE COLLECTION ROUTINE CONSISTS OF TWO MAJOR PROCESSES. THE FIRST PROCESS SEARCHES THE STRING DESCRIPTORS OF TEMPORARY VARIABLES, SIMPLE VARIABLES AND ARRAY VARIABLES. THE SEARCH LOCATES THE STRING DESCRIPTOR WHICH POINTS AT THE STRING DATA HIGHEST IN MEMORY. THE SECOND PROCESS MOVES THE STRING DATA

LOCATED IN THE SEARCH TO THE TOP OF MEMORY. THE CORRESPONDING DESCRIPTOR THEN HAS IT'S POINTER UPDATED SO THAT IT CORRECTLY POINTS TO THE STRING DATA AT IT'S NEW ADDRESS. THIS PROCESS CONTINUES UNTIL ALL ACTIVE STRINGS HAVE BEEN MOVED INTO ONE CONTIGUOUS BLOCK STARTING AT THE TOP OF MEMORY. DIAGRAM 2 SHOWS STRING SPACE BEFORE AND AFTER GARBAGE COLLECTION. DIAGRAM 3 SHOWS A FLOW CHART OF THE PROCESS.

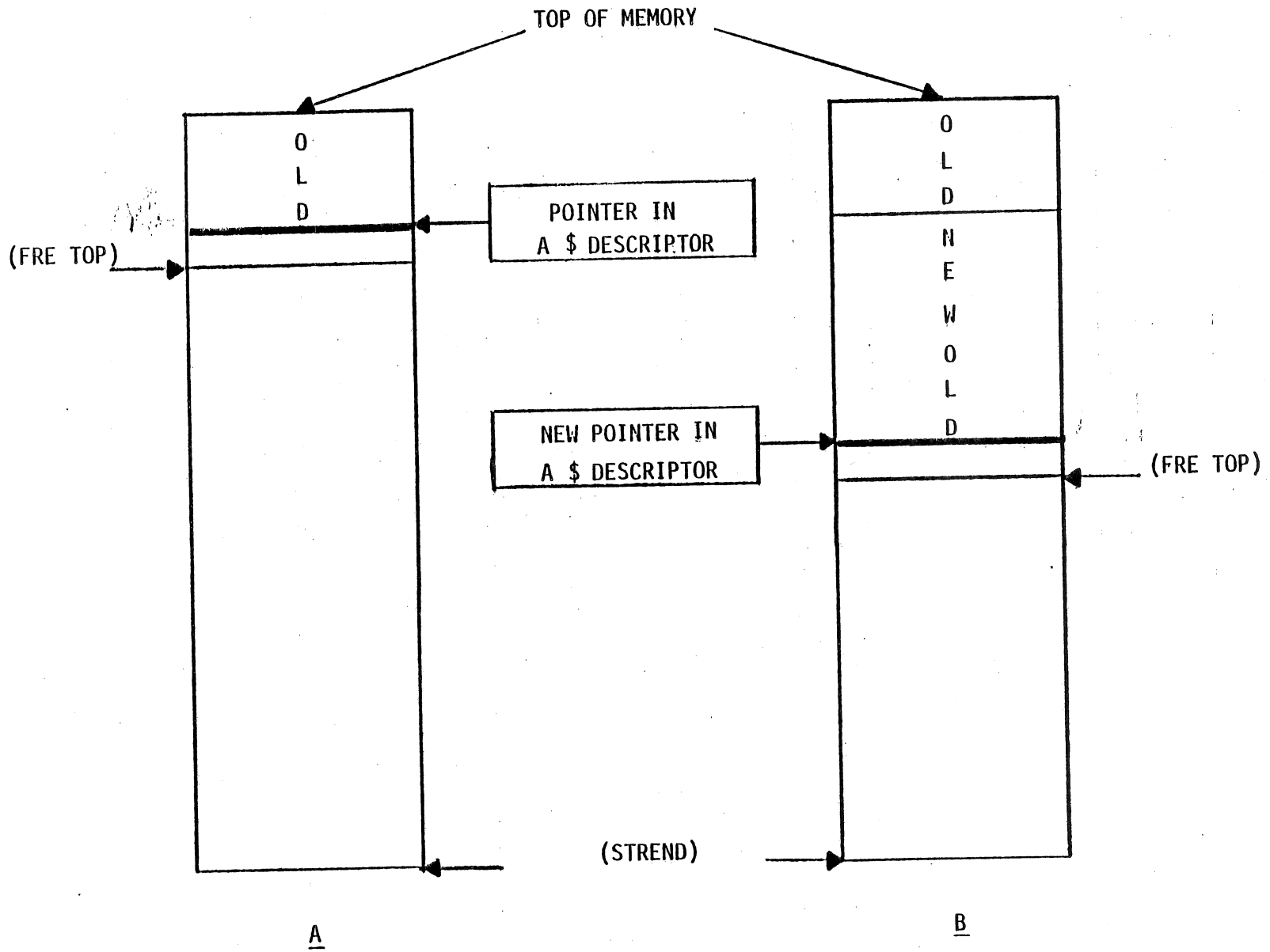

TOP OF MEMORY
O
L
D
(FRE TOP)
POINTER IN
A $ DESCRIPTOR
O
L
D
N
E
W
O
L
D
NEW POINTER IN
A $ DESCRIPTOR
(FRE TOP)
(STREND)
A
B

DIAGRAM 1

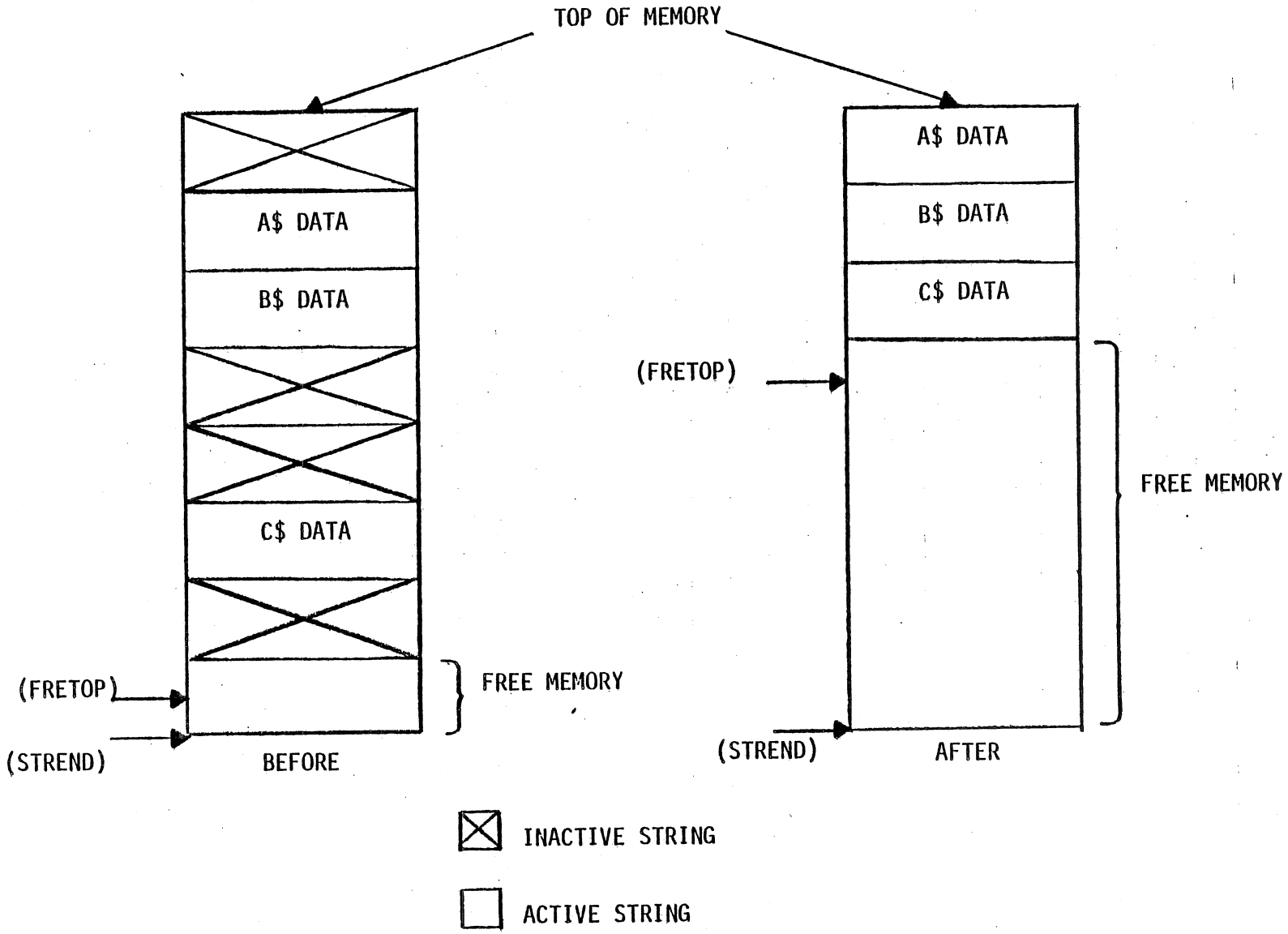

TOP OF MEMORY
A$ DATA
B$ DATA
C$ DATA
(FRETOP)
(STREND)
FREE MEMORY
BEFORE
A$ DATA
B$ DATA
C$ DATA
(FRETOP)
FREE MEMORY
(STREND)
AFTER
INACTIVE STRING
ACTIVE STRING

DIAGRAM 2

DIAGRAM 3A

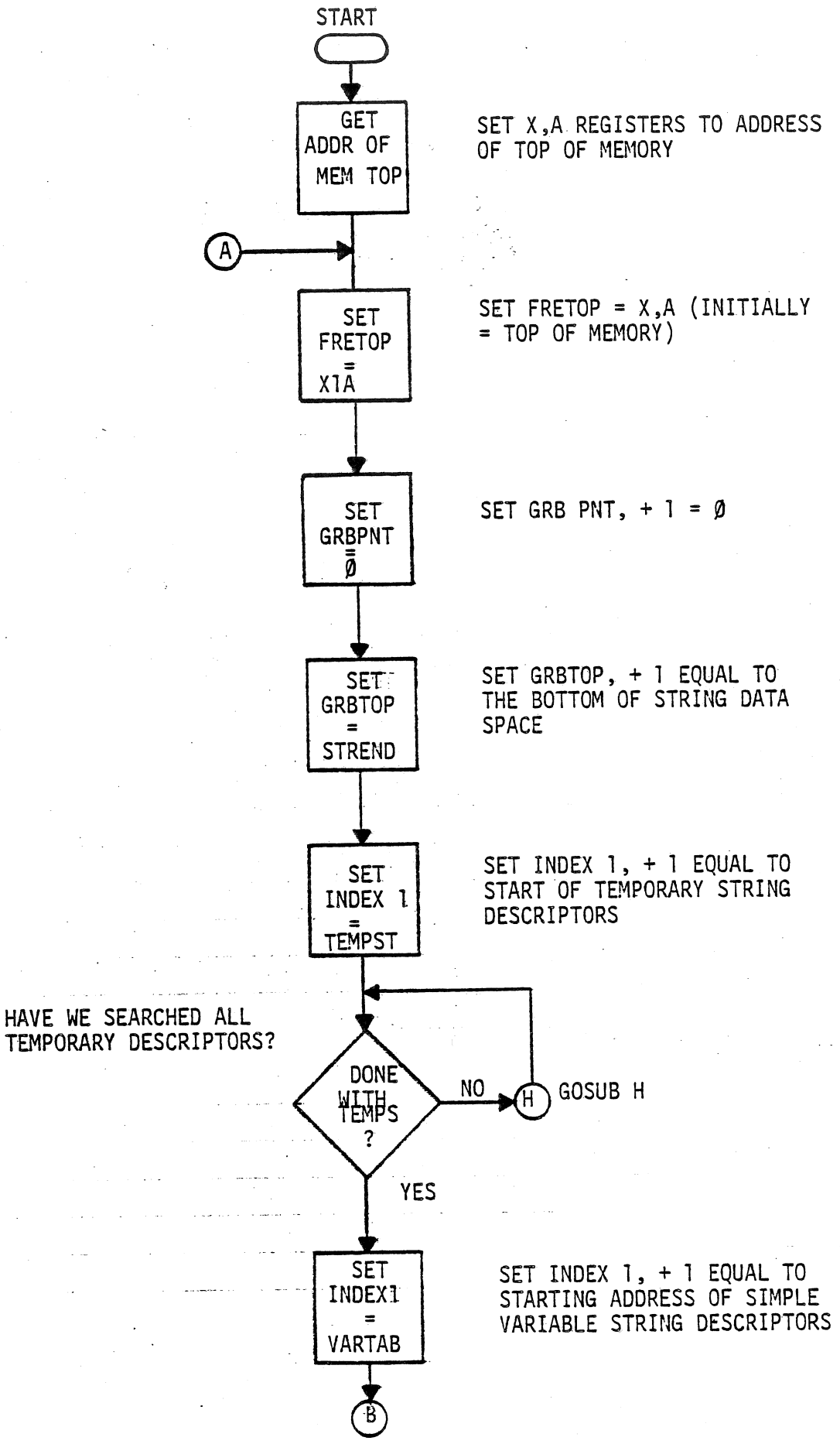

START
GET ADDR OF MEM TOP
SET X,A REGISTERS TO ADDRESS OF TOP OF MEMORY
A
SET FRETOP = X1A
SET FRETOP = X,A (INITIALLY = TOP OF MEMORY)
SET GRBPNT = Ø
SET GRB PNT, + 1 = Ø
SET GRBTOP = STREND
SET GRBTOP, + 1 EQUAL TO THE BOTTOM OF STRING DATA SPACE
SET INDEX 1 = TEMPST
SET INDEX 1, + 1 EQUAL TO START OF TEMPORARY STRING DESCRIPTORS
HAVE WE SEARCHED ALL TEMPORARY DESCRIPTORS?
DONE WITH TEMPS ?
NO
H
GOSUB H
YES
SET INDEX1 = VARTAB
SET INDEX 1, + 1 EQUAL TO STARTING ADDRESS OF SIMPLE VARIABLE STRING DESCRIPTORS
B

DIAGRAM 3B

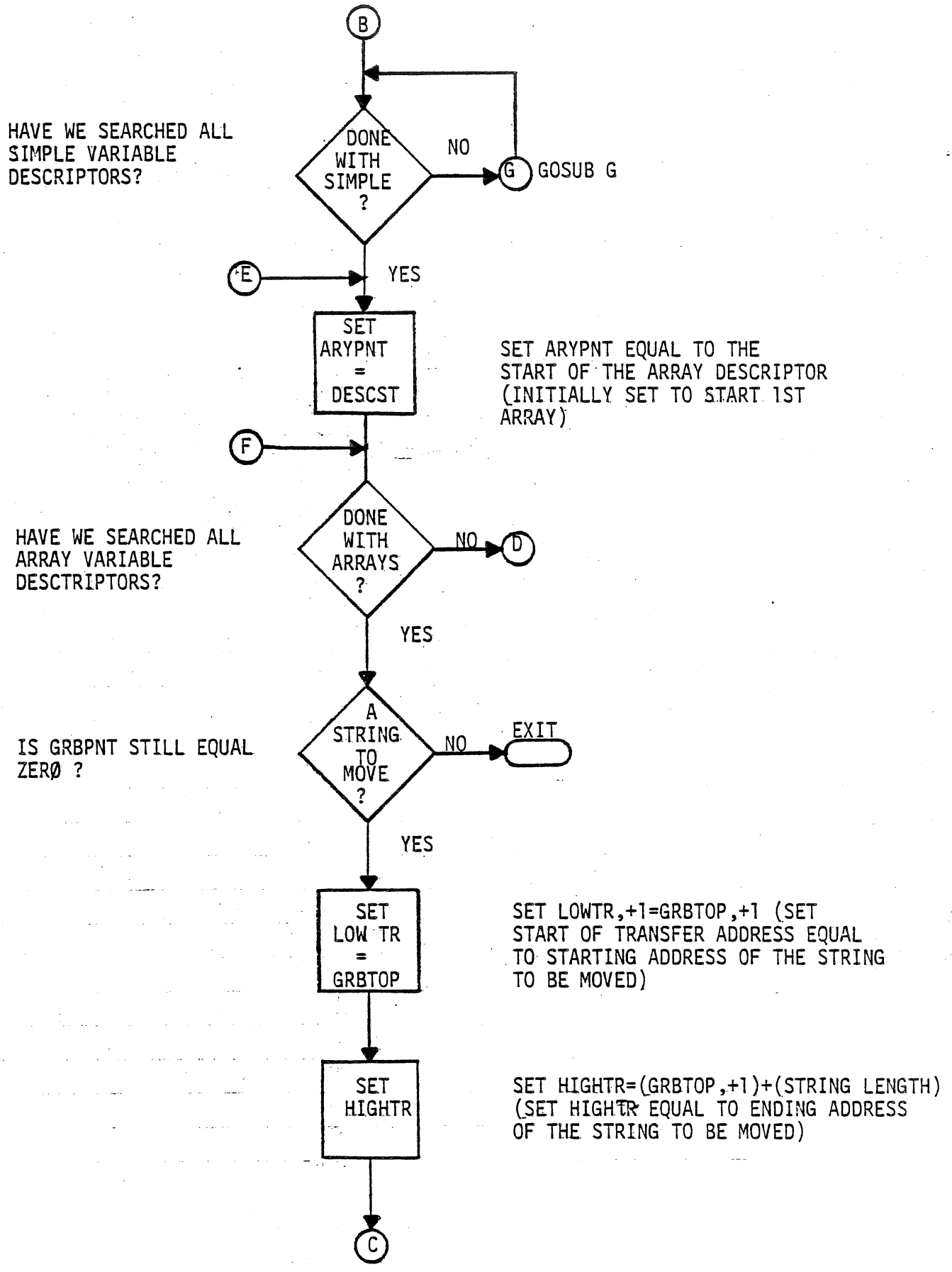
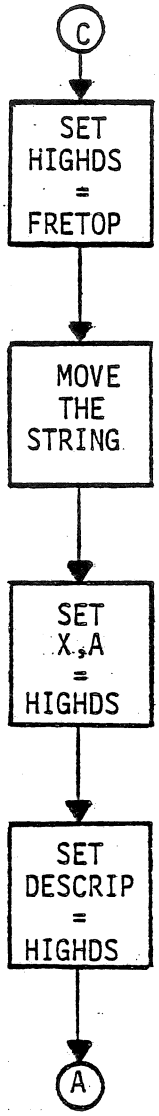

B
HAVE WE SEARCHED ALL SIMPLE VARIABLE DESCRIPTORS?
DONE WITH SIMPLE ?
NO
G
GOSUB G
E
YES
SET ARYPNT = DESCST
SET ARYPNT EQUAL TO THE START OF THE ARRAY DESCRIPTOR (INITIALLY SET TO START 1ST ARRAY)
F
HAVE WE SEARCHED ALL ARRAY VARIABLE DESCTRIPTORS?
DONE WITH ARRAYS ?
NO
D
YES
IS GRBPNT STILL EQUAL ZERØ ?
A STRING TO MOVE ?
NO
EXIT
YES
SET LOW TR = GRBTOP
SET LOWTR,+1=GRBTOP,+1 (SET START OF TRANSFER ADDRESS EQUAL TO STARTING ADDRESS OF THE STRING TO BE MOVED)
SET HIGHTR
SET HIGHTR=(GRBTOP,+1)+(STRING LENGTH) (SET HIGHTR EQUAL TO ENDING ADDRESS OF THE STRING TO BE MOVED)
C

DIAGRAM 3C

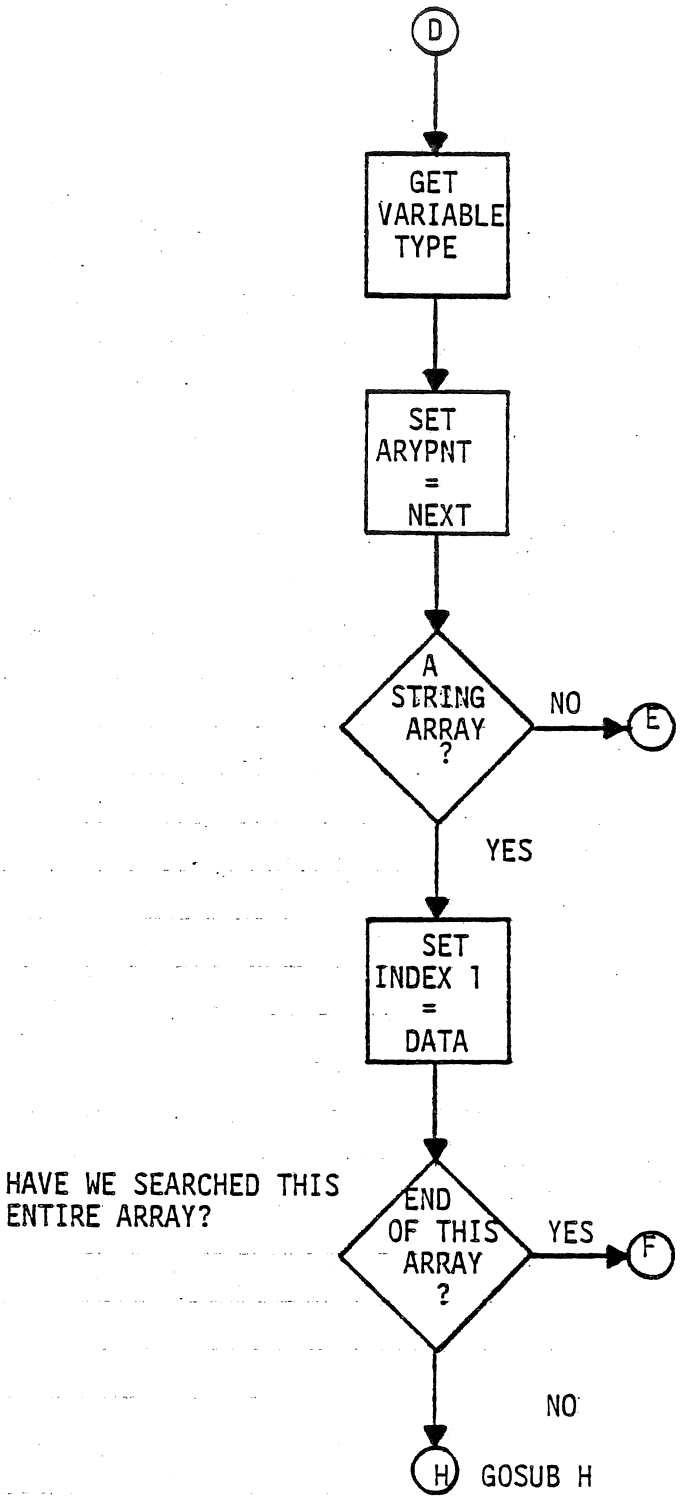


SET HIGHDS EQUAL TO THE ADDRESS OF THE BYTE JUST BELOW THE END OF THE LAST STRING MOVED (INITIALLY SET TO THE TOP OF MEMORY)

SET X,A REGISTERS EQUAL TO THE NEW STARTING ADDRESS OF THE STRING JUST MOVED

UPDATE THE DESCRIPTOR OF THE STRING JUST MOVED SO THAT IT POINTS TO THAT STRING AT IT'S NEW ADDRESS

DIAGRAM 3D



GET THE TWO BYTES CONTAINING THE VARIABLE NAME AND SAVE IN X,A

ADD TOTAL LENGTH OF THIS ARRAY TO IT'S STARTING ADDRESS AND SET ARYPNT EQUAL THE SUM

SET INDEX 1, + 1 EQUAL TO THE STARTING ADDRESS OF THE DESCRIPTORS IN THIS ARRAY

DIAGRAM 3E

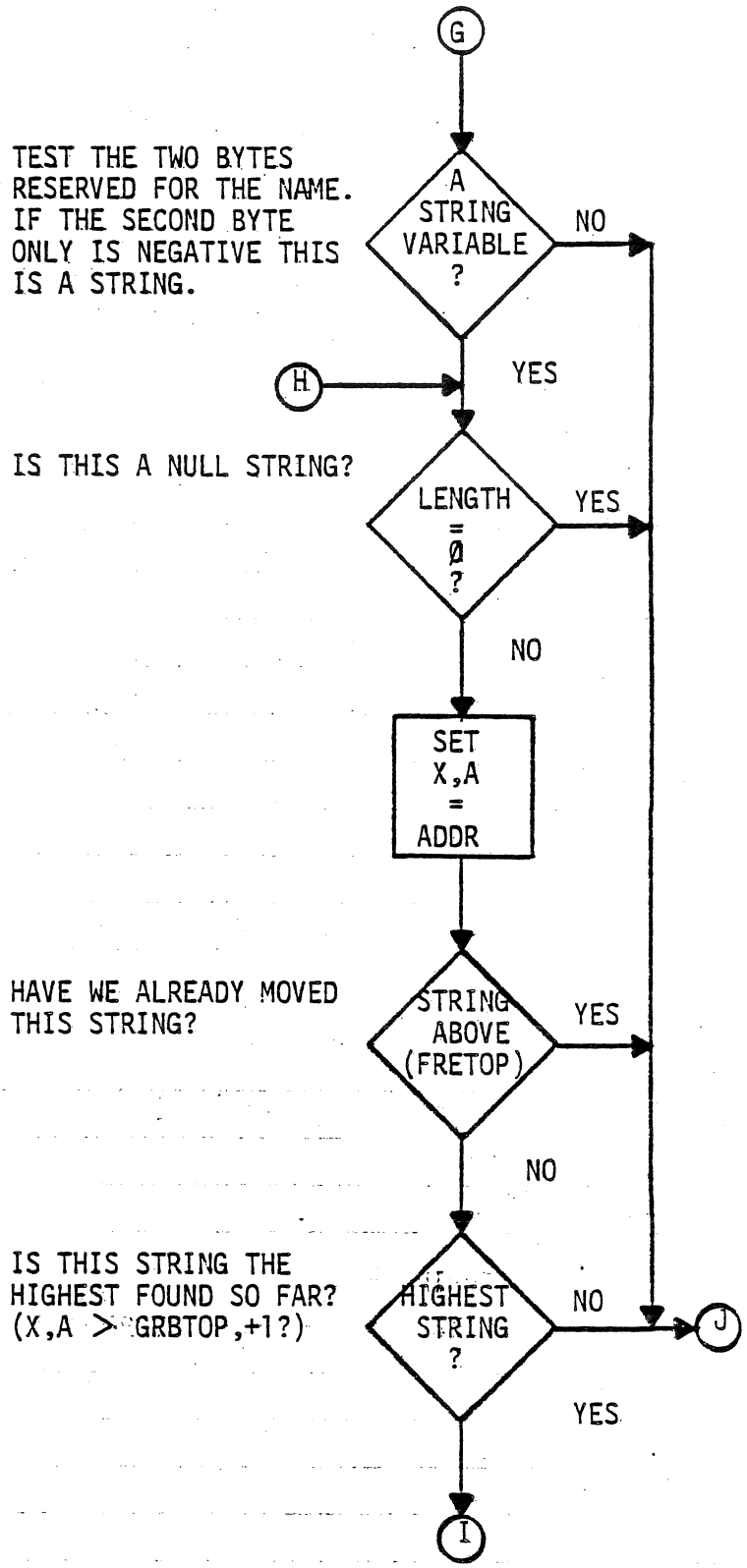
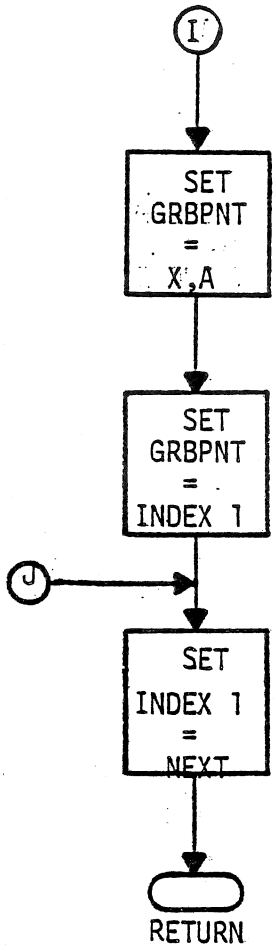

G
TEST THE TWO BYTES RESERVED FOR THE NAME. IF THE SECOND BYTE ONLY IS NEGATIVE THIS IS A STRING.
A STRING VARIABLE ?
NO
YES
H
IS THIS A NULL STRING?
LENGTH = Ø ?
YES
NO
SET X,A = ADDR
SET X,A EQUAL TO THE STARTING ADDRESS OF THIS STRING.
HAVE WE ALREADY MOVED THIS STRING?
STRING ABOVE (FRETOP)
YES
NO
IS THIS STRING THE HIGHEST FOUND SO FAR? (X,A > GRBTOP,+1?)
HIGHEST STRING ?
NO
J
YES
I

DIAGRAM 3F



SET GRBTOP, + 1 EQUAL TO THE ADDRESS OF THIS STRING

SET GRBPNT, + 1 EQUAL TO THE ADDRESS OF THIS STRING'S DESCRIPTOR

BUMP INDEX 1, + 1 SO THAT IT POINTS TO THE NEXT DESCRIPTOR IN MEMORY

OS-65U PACK FIELD COMMAND

THE FOLLOWING PAGES PROVIDE THE SOURCE FOR THE OS-65U PACK FIELD COMMAND. THIS COMMAND MAY BE USED IN ANY PRINT STATEMENTS. OUR ORIGINAL NEED EVOLVED OUT OF OS-DMS. WHEN PRINTING TO A DMS DATA FILE, ONE MUST ALWAYS "FILL OUT THE FIELD". USUALLY THIS IS ACCOMPLISHED BY ADDING SPACES TO THE LEFT SIDE OF THE STRING UNTIL THE STRING'S LENGTH EQUALS THE FIELD LENGTH. THE REASON FOR THIS IS TO PREVENT INTRA-RECORD GARBAGE. IF FOR EXAMPLE, ONE WERE TO PRINT A STRING 10 CHARACTERS LONG INTO A FIELD AND THEN PRINT A STRING 5 CHARACTERS LONG INTO THE SAME FIELD, THERE WOULD BE 5 "GARBAGE" CHARACTERS BEHIND THE 5 CHARACTER STRING.

TO PREVENT THIS ONE MUST ADD SPACES AS EXPLAINED ABOVE. THE SPACES ARE ADDED TO THE LEFT SIDE BECAUSE BASIC "THROWS AWAY" LEADING SPACES. IT CAN BE SEEN THAT A LOT OF STRING MANIPULATION WAS PREVIOUSLY REQUIRED TO HANDLE THIS PROBLEM. THE PACK FIELD COMMAND PROVIDES AN EASY MEANS OF "FILLING OUT THE FIELD". THE PACK COMMAND REQUIRES TWO ARGUMENTS, THE FIELD LENGTH (EXCLUDING THE CARRIAGE RETURN) AND A STRING EXPRESSION.

THE SYNTAX FOR THE PACK COMMAND IS:

```
PRINT & FL, STRING

      OR

PRINT # DEVICE NUMBER, & FL, STRING

      OR

PRINT % CHANNEL NUMBER,& FL, STRING
```

WHERE FL IS THE FIELD LENGTH AND STRING IS ANY STRING EXPRESSION, FL MUST BE AN INTEGER WITHIN THE RANGE Ø<=FL<= 255.

EACH VARIABLE TO BE PRINTED IN THE PACK FORM REQUIRES THE & FL, STRING EXPRESSION FORMAT. A PACK COMMAND MAY BE FOLLOWED BY A COMMA OR SEMICOLON. FOR EXAMPLE:

```
PRINT & 32,NA$,DA$

      OR

PRINT & 40,NA$;DA$
```

ARE BOTH "LEGAL" STATEMENTS.

THE PACK COMMAND SHOULD PROVE USEFUL IN FILE I/O AS WELL AS REPORT WRITING ON THE CRT OR PRINTER.

IMPLEMENTATION OF THE PACK CMD

THE PACK COMMAND CODE IS SHOWN ASSEMBLED TO RESIDE IN FRONT OF A BASIC PROGRAM. TO INTEGRATE THE PACK COMMAND:

1) ENTER THE SOURCE AS SHOWN.

2) DELETE LINES 15ØØ - 154Ø.

3) ASSEMBLE THE PACK CODE AND SAVE IT TO DISK.

4) USING LOAD32 OR LOAD48 CALL THE MACHINE CODE INTO OS-65U.

5) EXECUTE A GO COMMAND TO RETURN TO OS-65U.

6) ENTER A NEW 7Ø<CR>TO ALLOCATE SPACE FOR THE MACHINE CODE.

7) ENTER THE SUBROUTINE AT 63ØØØ AS SHOWN IN THE EXAMPLE BASIC PROGRAM.

8) SAVE THE PROGRAM TO DISK.

NOW WHENEVER THE PACK COMMAND IS DESIRED, LOAD THE FILE THAT WAS SAVED IN STEP 8 ABOVE. THEN ENTER YOUR PROGRAM AND SAVE IT TO DISK. TO ENABLE THE PACK COMMAND EXECUTE A GOSUB 63ØØØ - TO DISABLE THE COMMAND EXECUTE A GOSUB 63Ø5Ø. NOTE THAT THE PACK COMMAND MUST BE DISABLED BEFORE A PROGRAM THAT DOES NOT CONTAIN THE PACK COMMAND MACHINE CODE CAN BE RUN.

```
100 REM                PACK FIELD COMMAND
110 REM         (C) 1979 BY OHIO SCIENTIFIC, INC.
120 REM             ALL RIGHTS RESERVED
130 REM             WRITTEN BY R.WHITESEL 9/79
140 REM             THIS VERSION RIDES IN FRONT OF
150 REM             THE BASIC PROGRAM AT $6000.
160 REM
170 REM
180 REM   PACK SYNTAX :
190 REM
200 REM               PRINT & FIELD LENGTH, STRING EXPRESSION
210 REM               PRINT %CH,& FL,STRING
220 REM               PRINT #DV,& FL,STRING
230 REM
240 REM
250 REM IF FL IS THE DESIRED FIELD LENGTH & DA$ IS THE
260 REM NEW FIELD CONTENTS, THEN PRINT &FL, DA$ WILL
270 REM
280 REM 1) DETERMINE IF THE LENGTH OF DA$ IS >= FL
290 REM    IF SO, THEN DO A STANDARD PRINT
300 REM    ELSE, FL - LENGTH (DA$) LEADING SPACES
310 REM    WILL BE PRINTED FOLLOWED BY DA$.
320 REM
330 REM
500 REM GOSUB TO ENABLE THE PACK COMMAND
510 REM
520 GOSUB 63000
530 REM
540 REM
1000 REM A LITTLE DEMO
1010 REM
1020 FOR X=40 TO 60 STEP 4: FOR Y=1 TO 2000: NEXT Y
1030 X$="TEST"+STR$(X)
1040 PRINT "THE FIELD WIDTH IS";X
1050 PRINT "THE STRING IS ",X$
1060 PRINT: PRINT "PRINT &X,X$ LOOKS LIKE THIS :"
1070 PRINT: PRINT &X,X$;: T=POS(X): PRINT "  < POSTION = ";T: PRINT
1080 NEXT
1090 REM
2000 REM DISABLE THE PACK CMD
2010 REM
2020 GOSUB 63050
2030 END
2040 REM
2050 REM
2060 REM
2070 REM
63000 REM ENABLE THE PACK CMD
63010 REM
63020 POKE 2634,076: POKE 2635,000: POKE 2636,096
63030 RETURN
63040 REM
63050 REM DISABLE THE PACK CMD
63060 REM
63070 POKE 2634,032: POKE 2635,205: POKE 2636,012
63080 RETURN
63090 REM
```

```
.A

1000 0000          ;          PACK CMD OVERLAY CODE FOR OS-65U
1010 0000          ;          COPYRIGHT 1979 BY OHIO SCIENTIFIC, INC.
1020 0000          ;          ALL RIGHTS RESERVED
1030 0000          ;          IMPLEMENTED BY R. WHITESEL 8/79
1040 0000          ;
1050 0000          ;          THIS VERSION RESIDES IN FRONT OF
1060 0000          ;          A BASIC PROGRAM @ $6000...
1070 0000          ;
1080 0000          ; PACK PRINTS LEADING SPACES IN FRONT OF A STRING VARIABLE
1090 0000          ; UNTIL IT'S LENGTH EQUALS :
1100 0000          ; LEADING SPACES + LENGTH OF STRING IN ARG = ARG LENGTH
1110 0000          ; FOR EXAMPLE :
1120 0000          ; A$="TEST": L=20     THEN 'PRINT & L, A$' WOULD PRINT
1130 0000          ; 16 LEADING SPACES FOLLOWED BY THE CONTENTS OF A$ (i.e. "TEST")
1140 0000          ;
1150 0000          ;
1160 0000          ; SYNTAX :
1170 0000          ;
1180 0000          ;         PRINT & LENGTH, STRING EXPRESSION
1190 0000          ;         OR
1200 0000          ;         PRINT # DEVICE NUMBER, & LENGTH, STRING EXPRESSION
1210 0000          ;         OR
1220 0000          ;         PRINT % CHANNEL NUMBER, & LENGTH, STRING EXPRESSION
1230 0000          ;
1240 0000          ;
1250 0000          ; EQUATES :
1260 0000          ;
1270 0000          ; BASIC ROUTINES :
1280 0000          ;
1290 0000          FRMEVL=@06315    ; FORMULA EVALUATOR SUBROUTINE
1300 0000          OUTSPC=@05350    ; PRINTS A SPACE
1310 0000          CHRGET=@00300    ; RTS'S W CHR TYPE @ (TXTPTR+1)
1320 0000          CHRGOT=@00306    ; RTS'S W CHR TYPE @ (TXTPTR)
1330 0000          BASPRT=@05115    ; ENTRY POINT TO LET BASIC EXECUTE PRINT
1340 0000          GETBYT=@13030    ; RTS W BYTE IN X REG
1350 0000          FREFAC=@12440    ; FREES UP FAC
1360 0000          NEWCHR=@05062    ; DO NXT ENTRY POINT IN PRINT CODE
1370 0000          STRPRT=@05317    ; PRINTS STRING SET UP IN TEMP
1380 0000          TMERR:=@06310    ; TYPE MISMATCH ERROUT ROUTINE
1390 0000          CHKCOM=@07023    ; CHKS CHR @ (TXTPTR) IS A ','. IF / SN ERR
1400 0000          ;
1410 0000          ;
1420 0000          ; BASIC TMPS & POINTERS :
1430 0000          ;
1440 0000          VALTYP=@00016    ; 0=NUMERIC / 1=STRING
1450 0000          ANDTOK=@00203    ; '&' TOK
1460 0000          ;
1470 0000          ;
1480 0000          ;
1490 0000          ;
1500 0A4A                 *=@05112     ; OVERLAYS JSR FRMEVL IN PRINT CODE
1510 0A4A          ;
1520 0A4A 4C0060          JMP FILSTR       ; OVERLAY JSR FRMEVL
1530 0A4D          ;
1540 0A4D          ;
1550 6000                 *=$6000      ; RIDES IN FRONT OF BASIC PROGRAM
1560 6000          ;
1570 6000          ;
1580 6000 C983     FILSTR  CMP #ANDTOK      ; IS CHR A '&' TOK ?
```

```
1590 6002                ;
1600 6002 F006           ISFILL  BEQ ISFILL      ; YES- GO TO IT !
1610 6004 20CD0C                 JSR FRMEVL      ; EVAL FORMULA FOR BASIC
1620 6007 4C4D0A                 JMP BASPRT      ; AND LET BASIC PRINT IT
1630 600A 20C000         ISFILL  JSR CHRGET      ; GET NXT CHR
1640 600D                ;
1650 600D 201816                 JSR GETBYT      ; GET LENGTH ARG
1660 6010 8E4260                 STX SETLEN      ; SAVE LENGTH DESIRED
1670 6013 20130E                 JSR CHKCOM      ; ',' FOLLOWS ?
1680 6016 20CD0C                 JSR FRMEVL      ; EVALUATE THE FORMULA
1690 6019 240E                   BIT VALTYP      ; IS IT A STRING ?
1700 601B 3003                   BMI IS:STR      ; IT IS A STRING !
1710 601D 4CC80C                 JMP TMERR:      ; NUMERIC IS TYPE MISMATCH
1720 6020 202015         IS:STR  JSR FREFAC      ; FRE UP STRING DESCRIPTOR
1730 6023                ;
1740 6023 8D4360                 STA SAVLEN      ; SAVE LENGTH OF THIS STRING
1750 6026 CD4260                 CMP SETLEN      ; STRING LEN >= LEN DECLARED IN ARG ?
1760 6029 B00E                   BCS TOOBIG      ; YES- SO NO LEADIN SPACES
1770 602B AD4260                 LDA SETLEN      ; GET DIFFERENCE
1780 602E 38                     SEC
1790 602F ED4360                 SBC SAVLEN
1800 6032 AA                     TAX             ; CNT TO X-REG FOR OUT LOOP
1810 6033 20E80A         PRTSPC  JSR OUTSPC      ; PRINT A SPACE
1820 6036                ;
1830 6036 CA                     DEX             ; ALL DONE ?
1840 6037 D0FA                   BNE PRTSPC      ; LOOP TILL WE ARE !
1850 6039 AD4360         TOOBIG  LDA SAVLEN      ; GET STRING LENGTH BACK
1860 603C                ;
1870 603C 20D20A                 JSR STRPRT+3    ; PRINT THE STRING
1880 603F 4C320A                 JMP NEWCHR      ; LET BASIC CONTINUE WITH PRINT
1890 6042                ;
1900 6042                ;
1910 6042                ; LOCAL TEMPS :
1920 6042                ;
1930 6042 00             SETLEN  .BYTE $00       ; (LEN PASSED IN ARG)
1940 6043 00             SAVLEN  .BYTE $00       ; (LEN OF STR IN ARG)

.80
```