DECUS
PROGRAM LIBRARY

DECUS NO.	8-514
TITLE	ALPHA-NUMERIC DISPLAY PROGRAM
AUTHOR	Ralph Norman Haber
COMPANY	University of Rochester Rochester, New York
DATE	December 1971
SOURCE LANGUAGE	PAL and FORTRAN

ALPHA-NUMERIC DISPLAY PROGRAM

DECUS Program Library Write-up

DECUS NO. 8-514

A general description of this program was published by Ralph Norman Haber as "An alpha-numeric display program for a PDP-8", in Behavioral Research Methods and Instrumentation, 1971, 3, pp. 141-145. This interpretation differs in only small details from the published version as noted below. The published article will be helpful in understanding the purposes and strategies of the program.

The program requires a Driver, written in FORTRAN, a character display subroutine, and a button sensing subroutine. Each of these are described.

FORTRAN Driver Program

The purpose for the Driver is to operate the display subroutine after it has accepted all the parameter values for the next trial. The left side of Figure 1 provides the flow chart for the program, and Table 1 lists the statements. The following comments serve to annotate the program. (Note that the parameter codes given in the published article are slightly different from the version here. The present ones are listed in Table 2 below.)

The two Dimension statements create storage arrays for the character (IALP), durations (IDUR), length of strings (LENG), ISI (ISI), X locations (IXLO), Y locations (IYLO), number of strings (ISTR), number of cycles (IREP), and an array for storage of single value control characters (IARR). The printout from the symbol-print

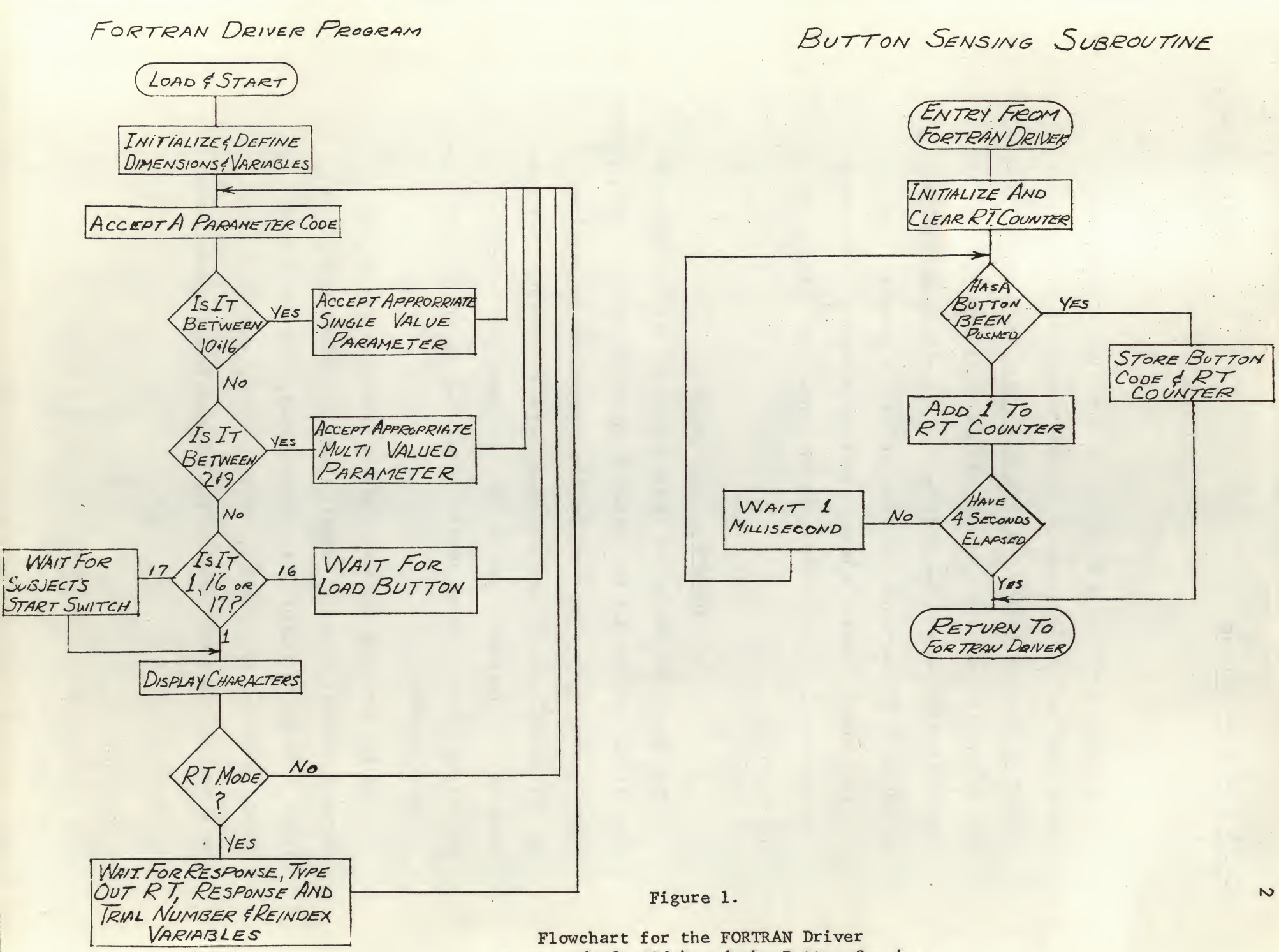

Figure 1.

Flowchart for the FORTRAN Driver Program (Left Side) and the Button Sensing Routine (Right Side)

```
      DIMENSION IALP(169),IDUR(16),LENG(16),ISI(16),IXLO(16),IYLO(16)
      DIMENSION ISTR(16),IREP(16),IARR(9)
      IRT=IRES=0
1; FORMAT (I)
2; FORMAT(/)
3; TYPE 2
      ACCEPT 1, INPUT
      IF(INPUT-10)29,20,32
32; IF(16-INPUT)29,29,20
4; DO 5 J1=1,100
      ACCEPT 1, IALP(J1)
      IF(IALP(J1)-99)5,3,5
5; CONTINUE
      GO TO 3
6; DO 7 J2=1,16
      ACCEPT 1, IDUR(J2)
      IF(IDUR(J2)-99)7,3,7
7; CONTINUE
      GO TO 3
8; DO 9 J3=1,16
      ACCEPT 1, ISTR(J3)
      IF(ISTR(J3)-99)9,3,9
9; CONTINUE
      GO TO 3
10; DO 11 J7=1,16
      ACCEPT 1, IREP(J7)
      IF(IREP(J7)-99)11,3,11
11; CONTINUE
      GO TO 3
12; DO 13 J8=1,16
      ACCEPT 1, LENG(J8)
      IF(LENG(J8)-99)13,3,13
13; CONTINUE
      GO TO 3
14; DO 15 J4=1,16
      ACCEPT 1, ISI(J4)
      IF(ISI(J4)-99)15,3,15
15; CONTINUE
      GO TO 3
16; DO 17 J5=1,16
      ACCEPT 1, IXLOC(J5)
      IF(IXLOC(J5)-99)17,3,17
17; CONTINUE
      GO TO 3
18; DO 19 J6=1,16
      ACCEPT 1, IYLOC(J6)
      IF(IYLOC(J6)-99)19,3,19
19; CONTINUE
      GO TO 3
20; ACCEPT 1,IARR(INPUT-9)
      GO TO 3
26; PAUSE 3584
      IF(IRES-3)26,3,26
27; PAUSE 3584
      IF(IRES-4)27,28,27
28; PAUSE 3520
      IF(IARR(6))30,3,30
29; GO TO(23,4,6,8,10,12,14,16,18,3,3,3,3,3,3,26,27),INPUT
      GO TO 3
30; PAUSE 3584
      TYPE 1, IARR(3),IRES,IRT
      IRT=IRES=0
      GO TO 3
      END
```

Table 1.

FORTRAN Driver Program
Statement Listing

attached to Table 1 and shows the locations in memory that these arrays are assigned by the FORTRAN compiler. The starting address of each of these arrays are used by the character subroutine as addresses to get each of the parameters as needed during each trial. The locations which store the reaction time and the subject's response code are initially set to zero.

The program begins operation with Statement 3, in which a single number (INPUT) is accepted from the typewriter. INPUT is used to specify which parameter is to be entered (See Table 2). If INPUT is less than 10 or greater than 15, the program jumps to Statement 29, which sorts out which one of the multivalued parameters is to be loaded. If INPUT is between 10 and 15, Statement 20 specifies the acceptance of any one of the six single-valued parameters. These are stored in the Dimension IARR, depending upon the value of INPUT. Thus, the single number following INPUT=10 would be stored in $IARR_1$, and the number following INPUT-15 would be stored in $ARRAY_6$.

Statement 29 handles the multivalued parameters and the operations parameters. If INPUT is 2 through 9, control is shifted to Statements 4 through 18, respectively. For example, if INPUT=2, the characters, defined numerically, are accepted. Statement 4 indicates that you will execute a loop up to 100 times, each time accepting a number which is stored in the dimension called IALP. After each character number is accepted, the IF statement tests to see whether the number was 99. If it was, the loop is terminated immediately, indicating that the last character has been entered. If it is not a 99, it continues

Table 2

A summary of the parameter codes.

Control Characters to Define Parameters for the Driver Program

1	=	initiate trial immediately
2	=	character names for up to 100 characters
4	=	number of strings
5	=	number of cycles
6	=	length of each string
7	=	interstimulus interval between each string
8	=	X location of each string
9	=	Y location of each string
10	=	vertical spacing between rows
11	=	horizontal spacing between characters on each row
12	=	trial number
13	=	vertical spacing between points within each character
14	=	horizontal spacing between each point within each character
15	=	reaction time mode control (0 = no reaction)
16	=	switch to permit additional parameter entry
17	=	wait for subject start switch to initiate trial

in the loop. When one hundred characters have been entered, or a 99 has been typed, the program returns to Statement 3 and waits for another parameter. If INPUT=3, the program will jump to Statement 6, where it will accept up to 16 durations, or until it encounters a 99. The logic for the Statements 4 through 18 are identical.

After all the parameters have been entered, then typing INPUT=1 causes a jump to the display subroutine and immediate beginning of the display. INPUT=17 causes a jump to a button sensing routine which waits for the subject to press a start button before the display occurs. INPUT=16 causes a wait for a button press which, when pressed, permits you to enter further parameters. This procedure is used when some of the trial parameters are on paper tape, and some are to be entered from the typewriter. This might be needed during pretesting, when the value of the durations might not be known yet.

Statement 28, causing the display to commence, is reached either by typing INPUT=2 or INPUT=17 (and the subject has pressed the start button). After the trial ends, $IARR_6$ is tested to see if the subject's reaction and his reaction time is to be recorded. If $IARR_6$ is non zero, the button sensing subroutine is entered, which counts the number of milliseconds that have elapsed since arriving there, and waits for the subject to respond with one of two response buttons. When he responds the reaction choice and the reaction time are stored. Then the driver program types these out, along with the trial number ($IARR_3$). This ends the trial. If $IARR_6$=0, program control returns for the beginning of the next trial, so that the parameter values can be modified.

In this version of the program the spacing between characters and the spacing between points within characters are not preset, but must be entered before the first trial. Thus parameters 10, 11, 13, and 14 must be typed. These need only be entered once before the first trial, and need not be altered unless that is an experimental parameter. Parameter 12 for the trial number (stored as $IARR_3$), should be entered if you are printing reaction times. It is possible to rewrite the program with the trial number simply incremented from the beginning of the session. I use this version because it provides an output which is uniquely linked to the parameter listing.

Needless to say, it is not necessary to retype any parameter between one trial and the next that is not to be changed on the next trial. Only those parameters which are changing have to be reentered.

If any modifications are made in the program, several suggestions might be of some help. The order in which the dimension statements are written determines the locations they are assigned by the FORTRAN compiler and given in the symbol print printout. If any of those are changed at all, then it is necessary to alter the appropriate values in the display subroutine. Thus, while we have made many changes in this driver program, we have not made any changes in the dimension statements at all. This accounts, for example, for why the DO loop in statement 4 only accepts 100 characters even though we have storage room for 169. We do not need the other 69, but if size of the array is reduced, then you will have to alter about a dozen different addresses in the character subroutine.

This program operates with the paper tape and typewriter input, and typewriter output for the reaction time. In this version we are using, an ACCEPT statement causes an echo on the teleprinter, so that all values are typed out as they are loaded. If you wish to suppress the echo, then replace the ACCEPT with READ 1, the 1 defining teletype and slow speed paper tape inputs.

If you plan to use the DECtape input, then each of the ACCEPT or READ 1 Statements must be changed to a READ 3 Statement, and the first Dimension Statement preceeded with a "Define Tape" Statement. When this is done, however, the FORTRAN Operating System uses nearly an additional 1,000 locations to handle the DECtape controls. This does not leave enough room in the Operating System to store the character subroutine. Consequently, it is not possible to use DECtape inputs for parameters with only 4K of memory. In an 8K system, there is no problem at all. If you want to use DECtape but have only 4K it is probably possible to rewrite the character program in several parts. For example, the first part might be in core and would get the particular values for the next trial. Those values would be stored, and then that part of the program overlaid with the section which handles the display and indexing. There would be no need to store in core the 35 bits for each character. All one would need is to move into core those characters that are actually to be displayed on the current trial.

This brief description of the driver program plus the flowchart and listing should permit its use on any PDP-8 family computer, utilizing only 4K of memory.

Display Subroutine

The display subroutine occupies from 6,000 to 6,477 (about 300_{10} locations), though there are a few holes here and there that could be filled with other subroutines. The 35 bits for each of the 64 possible characters are stored in 6,500 to 6,777. A flowchart for this subroutine is attached in Figure 2 and an anotated symbolic listing of the subroutine is attached in Table 3.

This subroutine uses the extended arithmetic element of the PDP-8 to compute the horizontal and vertical spacing between characters. See locations 6,055 - 6,064, and 6,065 - 6,074. If you do have the extended arithmetic unit on your PDP-8, then that portion of the program would have to be replaced with either a software multiplication procedure, or with a more direct procedure for converting the X and Y axis values to coordinate values.

This display subroutine uses a 34D control and a display system with a Tektronics 503 scope. There are two places where the display instructions are used, one in the main part of the program for displaying characters (locations 6,230 and 6,233), and another in a subroutine near the end for locating the position in which the electron beam rests between assigned positions (locations 6,321 and 6,324). If your control system uses a different I/O instruction, then you would have to make appropriate changes.

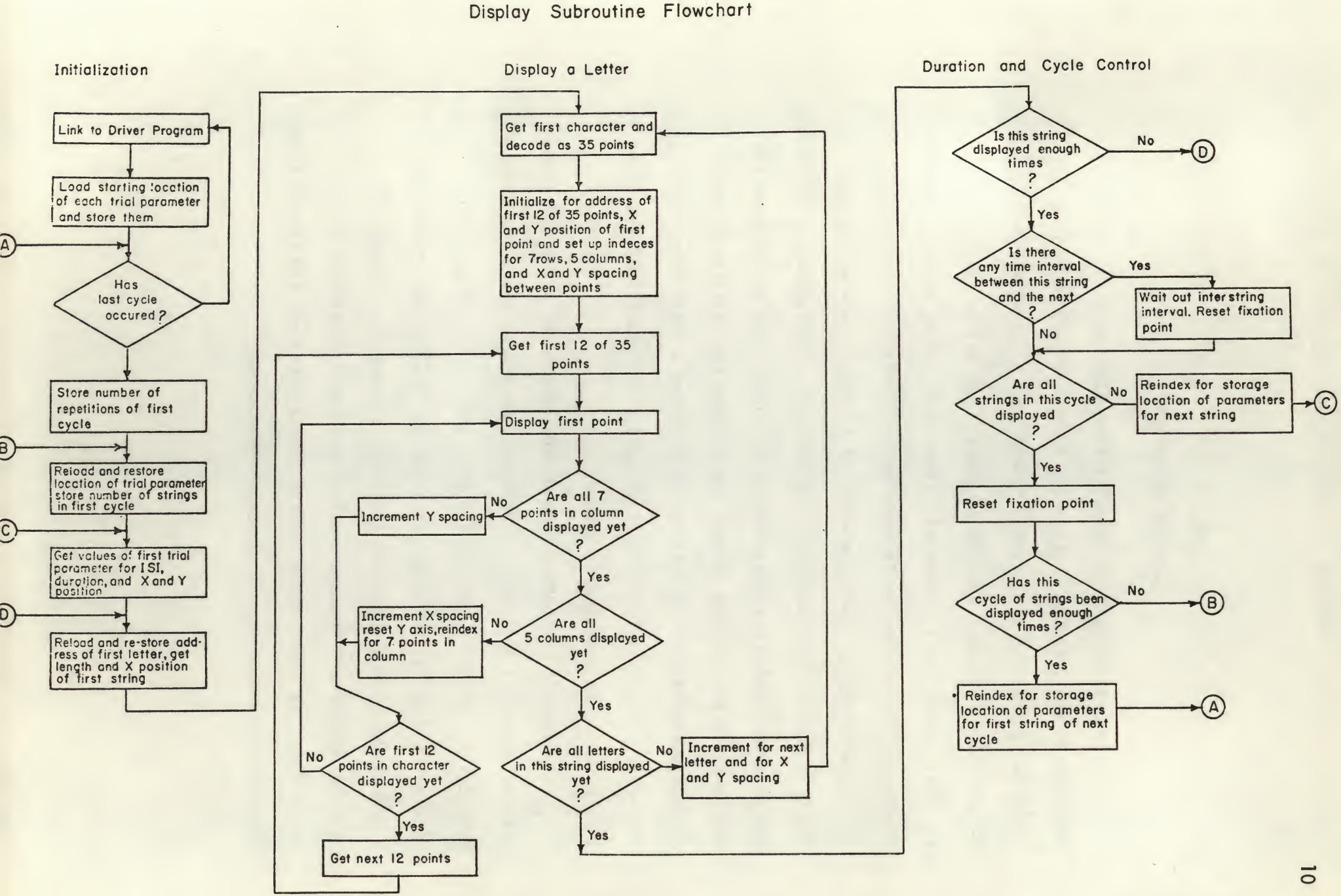

Figure 2.

Flowchart for the Character Display Subroutine

Table 3.

Character Display Subroutine

Ralph Norman Haber

University of Rochester

```
          *6000
6000  0000  DISPLAY, 0          SUBROUTINE LINK
6001  1325  TAD DUR             LOAD ADDRESS OF FIRST DURATION
6002  3326  DCA PAT             STORE IT
6003  1327  TAD XLOC            LOAD ADDRESS OF FIRST X LOCATION
6004  3330  DCA XLIN            STORE IT
6005  1331  TAD YLOC            LOAD ADDRESS OF FIRST Y LOCATION
6006  3332  DCA YLIN            STORE IT
6007  1333  TAD LENGTH          LOAD ADDRESS OF LENGTH OF FIRST STRING
6010  3334  DCA EXTENT          STORE IT
6011  1335  TAD NO              LOAD ADDRESS OF NUMBER OF STRINGS
6012  3336  DCA STRING          STORE IT
6013  1337  TAD LETTER          LOAD ADDRESS OF NAME OF FIRST CHARACTER
6014  3340  DCA ALPHA           STORE IT
6015  1341  TAD ISI             LOAD ADDRESS OF ISI
6016  3342  DCA HENRY           STORE IT
6017  1343  TAD REPET           LOAD ADDRESS OF NUMBER OF REPETITION CYCLES
6020  3344  DCA AGAIN           STORE IT
6021  1744  TOP, TAD I AGAIN    LOAD NUMBER OF REPETITION CYCLES
6022  7041  CIA                 MAKE IT NEGATIVE
6023  7450  SNA                 ARE THERE ANY MORE CYCLES?
6024  5600  JMP I DISPLAY       NO, EXIT FROM SUBROUTINE
6025  3756  DCA I CYCLEI        YES, STORE IT ON NEXT PAGE OF SUBROUTINE
6026  1326  BEGIN, TAD PAT      LOAD ADDRESS OF FIRST DURATION
6027  3345  DCA EUNICE          STORE IT
6030  1330  TAD XLIN            LOAD ADDRESS OF FIRST X LOCATION
6031  3346  DCA XPOS            STORE IT
6032  1332  TAD YLIN            LOAD ADDRESS OF FIRST Y LOCATION
6033  3347  DCA YPOS            STORE IT
6034  1334  TAD EXTENT          LOAD ADDRESS OF LENGTH OF FIRST STRING
6035  3350  DCA NUMBER          STORE IT
6036  1736  TAD I STRING        LOAD NUMBER OF STRINGS IN FIRST CYCLE
6037  7041  CIA                 MAKE IT NEGATIVE
6040  3751  DCA I SARAHI        STORE IT ON NEXT PAGE
6041  1340  TAD ALPHA           LOAD ADDRESS OF FIRST CHARACTER
6042  3352  DCA JOAN            STORE IT
6043  1342  TAD HENRY           LOAD ADDRESS OF FIRST ISI
6044  3353  DCA GEORGE          STORE IT
6045  1754  TAD I XSHIFT        TRANSFER VALUE OF X SPACING TO NEXT
6046  3755  DCA I XGAPI                 PAGE OF SUBROUTINE
6047  1753  BOW, TAD I GEORGE   LOAD ISI
6050  7041  CIA                 MAKE IT NEGATIVE
6051  3757  DCA I MABELI        STORE IT ON NEXT PAGE
6052  1745  TAD I EUNICE        LOAD DURATION
6053  7041  CIA                 MAKE IT NEGATIVE
6054  3760  DCA I TIMEI         STORE IT ON NEXT PAGE
6055  1754  TAD I XSHIFT        LOAD X SPACING BETWEEN LETTERS
```

Table 3, Continued

```
6056  7421  7421
6057  1746  TAD I XPOS          STORE IN MQ
6060  3262  DCA .+2             LOAD STARTING X POSITION
6061  7405  7405                STORE AHEAD 2 LOCATIONS
6062  0000  0000                MULTIPLY TOGETHER
6063  7701  7701                STORAGE LOCATION FOR MULTIPLY INSTRUCTIONS
6064  3361  DCA XBEGIN          CLEAR AC AND LOAD PRODUCT INTO AC
6065  1762  TAD I YSHIFT        STORE THE PRODUCT
6066  7421  7421
6067  1747  TAD I YPOS
6070  3272  DCA .+2             REPEAT PROCESS FOR Y POSITION
6071  7405  7405
6072  0000  0000
6073  7701  7701
6074  3763  DCA I YSTATI
6075  1352  WOW, TAD JOAN       LOAD ADDRESS OF FIRST WORD
6076  3364  DCA ALAN            STORE IT
6077  1750  TAD I NUMBER        LOAD LENGTH OF FIRST STRING
6100  7041  CIA                 MAKE IT NEGATIVE
6101  3765  DCA I STEVEI        STORE IT ON NEXT PAGE
6102  1361  TAD XBEGIN          LOAD STARTING X AXIS
6103  3766  DCA I XSTATI        STORE ON NEXT PAGE
6104  1764  START, TAD I ALAN   LOAD FIRST CHARACTER
6105  0367  AND MASK            LOOK AT LAST BIT
6106  1370  TAD DICK            ADD 6500 TO GET ADDRESS OF FIRST 12 POINTS
6107  3371  DCA SAM             STORE IT
6110  1771  TAD I SAM           LOAD FIRST 12 POINTS
6111  3372  DCA TOM1            STORE IT
6112  1371  TAD SAM             LOAD ADDRESS OF FIRST 12 POINTS
6113  1375  TAD DALE            ADD 100 TO ADDRESS OF 2ND 12 POINTS
6114  3371  DCA SAM             STORE ADDRESS
6115  1771  TAD I SAM           LOAD 2ND 12 POINTS
6116  3373  DCA TOM2            STORE THEM
6117  1371  TAD SAM             LOAD ADDRESS OF 2ND 12 POINTS
6120  1375  TAD DALE            ADD 100 TO GET ADDRESS OF 3RD 12 POINTS
6121  3371  DCA SAM             STORE IT
6122  1771  TAD I SAM           LOAD 3RD 12 POINTS
6123  3374  DCA TOM3            STORE THEM
6124  5776  JMP I NEXTI         JUMP TO NEXT PAGE
6125  7306  DUR, 7306           LOCATION WHERE FORTRAN ARRAY FOR
                                      DURATION BEGINS
6126  0000  PAT, 0
6127  7226  XLOC, 7226          LOCATION WHERE FORTRAN ARRAY FOR
6130  0000  XLIN, 0                  X LOCATION BEGINS
6131  7206  YLOC, 7206          LOCATION WHERE FORTRAN ARRAY FOR
6132  0000  YLIN, 0                  Y LOCATION BEGINS
6133  7266  LENGTH, 7266        LOCATION WHERE FORTRAN ARRAY FOR
6134  0000  EXTENT, 0                LENGTH OF STRINGS BEGINS
6135  7166  NO, 7166
6136  0000  STRING, 0
6137  7326  LETTER, 7326        LOCATION WHERE FORTRAN ARRAY FOR
6140  0000  ALPHA, 0                 NAME OF CHARACTERS BEGINS
6141  7246  ISI, 7246           LOCATION WHERE FORTRAN ARRAY FOR
6142  0000  HENRY, 0                 ISI BEGINS
6143  7146  REPET, 7146         LOCATION WHERE FORTRAN ARRAY FOR
6144  0000  AGAIN, 0                 NUMBER OF CYCLES BEGINS
6145  0000  EUNICE, 0
6146  0000  XPOS, 0
6147  0000  YPOS, 0
6150  0000  NUMBER, 0
6151  6362  SARAHI, SARAH
```

```
6152  0000  JOAN, 0
6153  0000  GEORGE, 0
6154  7136  XSHIFT, 7136        LOCATION OF FORTRAN VARIABLE FOR
6155  6352  XGAPI, XGAP               VALUE OF X SPACING BETWEEN POINTS
6156  6450  CYCLEI, CYCLE             BEGINS
6157  6356  MABELI, MABEL
6160  6354  TIMEI, TIME
6161  0000  XBEGIN, 0
6162  7135  YSHIFT, 7135        LOCATION WHERE FORTRAN ARRAY FOR
6163  6333  YSTATI, YSTART            VALUE OF Y SPACING BETWEEN POINTS
6164  0000  ALAN, 0                   BEGINS
6165  6350  STEVEI, STEVE
6166  6331  XSTATI, XSTART
6167  0077  MASK, 0077
6170  6500  DICK, 6500
6171  0000  SAM, 0
6172  0000  TOM1, 0
6173  0000  TOM2, 0
6174  0000  TOM3, 0
6175  0100  DALE, 0100
6176  6200  NEXTI, NEXT
            *6200
6200  1327  NEXT, TAD FRANK       LOAD LOCATION OF FIRST 12 POINTS
6201  3330  DCA HARRY             STORE IT
6202  1331  TAD XSTART             INDEX FOR X AXIS FOR FIRST POINT
6203  3332  DCA XAXIS              STORE IT
6204  1333  TAD YSTART             INDEX FOR Y AXIS OF FIRST POINT
6205  3334  DCA YAXIS              STORE IT
6206  1335  TAD P14                INDEX FOR 12 POINTS
6207  3336  DCA BILL               STORE
6210  1337  TAD C5                 INDEX FOR 5 COLUMNS
6211  3340  DCA PETE               STORE
6212  1341  TAD R7                 INDEX FOR 7  ROWS
6213  3342  DCA FRED               STORE
6214  1743  TAD I XROW             LOAD INDEX FOR HORIZONTAL SPACE BETWEEN POINTS
6215  3345  DCA XSPACE             STORE
6216  1744  TAD I YHIGH            LOAD INDEX FOR VERTICAL SPACE BETWEEN POINTS
6217  3346  DCA YSPACE             STORE
6220  1730  HERE, TAD I HARRY      LOAD FIRST 12 POINTS
6221  3347  DCA JEFF               STORE
6222  1347  NOW, TAD JEFF          RELOAD
6223  7004  RAL                    SHIFT 1 BIT TO LEFT
6224  3347  DCA JEFF               STORE
6225  7420  SNL                    IS LINK = 1?
6226  5235  JMP OK                 NO, DO NOT DISPLAY
6227  1332  TAD XAXIS              YES, LOAD X AXIS
6230  6053  6053                   PUT IT IN X BUFFER
6231  7200  CLA                    CLEAR AC
6232  1334  TAD YAXIS              LOAD Y AXIS
6233  6067  6067                   PUT IT IN Y BUFFER, AND DISPLAY POINT
6234  7200  CLA                    CLEAR AC
6235  2342  OK, ISZ FRED           IS FIRST COLUMN OF 7 POINTS DONE?
6236  5240  JMP .+2                NO
6237  5251  JMP THIS               YES
6240  1334  TAD YAXIS              LOAD Y AXIS
6241  1346  TAD YSPACE             INCREMENT ONE UNIT
6242  3334  DCA YAXIS              RESTORE
6243  2336  BUT, ISZ BILL          ARE FIRST 12 POINTS DONE?
6244  5222  JMP NOW                NO
6245  2330  ISZ HARRY              YES    REINDEX FOR NEXT 12 POINTS
```

Table 3, Continued

```
6246  1335  TAD P14
6247  3336  DCA BILL
6250  5220  JMP HERE
6251  2340  THIS, ISZ PETE
6252  5254  JMP .+2
6253  5264  JMP MORE
6254  1332  TAD XAXIS
6255  1345  TAD XSPACE
6256  3332  DCA XAXIS
6257  1333  TAD YSTART
6260  3334  DCA YAXIS
6261  1341  TAD R7
6262  3342  DCA FRED
6263  5243  JMP BUT
6264  2350  MORE, ISZ STEVE
6265  5267  JMP .+2
6266  5274  JMP COUNT
6267  2751  ISZ I ALANI
6270  1331  TAD XSTART
6271  1352  TAD XGAP
6272  3331  DCA XSTART
6273  5753  JMP I STARTI
6274  2354  COUNT, ISZ TIME
6275  5755  JMP I WOWI
6276  1356  TAD MABEL
6277  7440  SZA
6300  5302  JMP .+2
6301  5312  JMP SAID
6302  3357  DCA CORA
6303  4316  JMS END
6304  1360  LOOP, TAD M260
6305  3361  DCA LOW
6306  2361  ISZ LOW
6307  5306  JMP .-1
6310  2357  ISZ CORA
6311  5304  JMP LOOP
6312  2362  SAID, ISZ SARAH
6313  5763  JMP I HOPEI
6314  4316  JMS END
6315  5764  JMP I MAYBEI
6316  0000  END, 0
6317  7200  CLA
6320  1765  TAD I XEND
6321  6053  6053
6322  7200  CLA
6323  1766  TAD I YEND
6324  6067  6067
6325  7200  CLA
6326  5716  JMP I END
6327  6172  FRANK, TOM1
6330  0000  HARRY, 0
6331  0000  XSTART, 0
6332  0000  XAXIS, 0
6333  0000  YSTART, 0
6334  0000  YAXIS, 0
6335  7764  P14, 7764
6336  0000  BILL, 0
6337  7773  C5, 7773
6340  0000  PETE, 0
6341  7771  R7, 7771
```

```
CONTINUE WITH NEXT 12 POINTS
ARE ALL 5 COLUMNS DONE?
NO
YES

REINDEX FOR NEXT COLUMN


RESET Y AXIS TO BOTTOM

REINDEX FOR 7 ROWS
CHECK FOR 12 POINTS
ARE ALL CHARACTERS IN FIRST STRING DONE?
NO
YES
ADD 1 TO ADDRESS OF CHARACTER STRING

MOVE X AXIS OVER BY SPACE BETWEEN
     CHARACTER
DO MORE LETTERS
IS FIRST STRING DISPLAYED ENOUGH TIMES?
NO, DO IT SOME MORE
YES, LOAD ISI
IS THERE ANY TIME BETWEEN THIS STRING
YES          AND NEXT?
NO, GO TO NEXT STRING
RESTORE ISI
SUBROUTINE TO LOCATE FIXATION POINT
     DURING ISI

LOOP TO WAIT 1 MILLISECOND

HAVE WE WAITED LONG ENOUGH IN ISI?
NO, WAIT SOME MORE
YES, HAVE ALL STRINGS BEEN DISPLAYED ONCE?
NO, DO MORE STRINGS
YES, RESET FIXATION POINT
CHECK TO SEE IF ALL STRINGS ARE TO BE
     RECYCLED.


SUBROUTINE TO LOCATE FIXATION POINTS
```

Table 3, Continued

```
6342  0000  FRED, 0
6343  7141  XROW, 7141                    X SPACING BETWEEN CHARACTERS
6344  7140  YHIGH, 7140                   Y SPACING BETWEEN ROWS
6345  0000  XSPACE, 0
6346  0000  YSPACE, 0
6347  0000  JEFF, 0
6350  0000  STEVE, 0
6351  6164  ALANI, ALAN
6352  0000  XGAP, 0
6353  6104  STARTI, START
6354  0000  TIME, 0
6355  6075  WOWI, WOW
6356  0000  MABEL, 0
6357  0000  CORA, 0
6360  7300  M260, 7300
6361  0000  LOW, 0
6362  0000  SARAH, 0
6363  6400  HOPEI, HOPE
6364  6411  MAYBEI, MAYBE
6365  7145  XEND, 7145                    X COORDINATE OF FIXATION POINT
6366  7144  YEND, 7144                    Y COORDINATE OF FIXATION POINT
6367  4772  JMS I DISPLI
6370  7402  HLT
6371  5367  JMP .-2
6372  6000  DISPLI, DISPLAY
            *6400
6400  2640  HOPE, ISZ I NUMBRI
6401  2641  ISZ I XPOSI
6402  2642  ISZ I YPOSI
6403  2643  ISZ I EUNICI
6404  2644  ISZ I GEORGI                  REINDEX FOR NEXT STRING
6405  1645  TAD I BTI
6406  7001  IAC
6407  3646  DCA I JOANI
6410  5647  JMP I BOWI
6411  2250  MAYBE, ISZ CYCLE              IS CURRENT CYCLE ALL DONE?
6412  5651  JMP I BEGINI                  NO, DO WHOLE DISPLAY CYCLE OVER AGAIN
6413  1643  TAD I EUNICI                  YES, RESET VALUES FOR NEXT CYCLE
6414  7001  IAC
6415  3652  DCA I PATI
6416  1641  TAD I XPOSI
6417  7001  IAC
6420  3653  DCA I XLINI
6421  1642  TAD I YPOSI
6422  7001  IAC
6423  3654  DCA I YLINI
6424  1640  TAD I NUMBRI
6425  7001  IAC
6426  3655  DCA I EXTNTI
6427  2656  ISZ I STRNGI
6430  1646  TAD I JOANI
6431  7001  IAC
6432  3657  DCA I ALPHAI
6433  1644  TAD I GEORGI
6434  7001  IAC
6435  3660  DCA I HENRYI
6436  2661  ISZ I AGAINI
6437  5662  JMP I TOPI
6440  6150  NUMBRI, NUMBER
6441  6146  XPOSI, XPOS
```

```
6442  6147  YPOSI, YPOS
6443  6145  EUNICI, EUNICE
6444  6153  GEORGI, GEORGE
6445  6164  BTI, ALAN
6446  6152  JOANI, JOAN
6447  6047  BOWI, BOW
6450  0000  CYCLE, 0
6451  6026  BEGINI, BEGIN
6452  6126  PATI, PAT
6453  6130  XLINI, XLIN
6454  6132  YLINI, YLIN
6455  6134  EXTNTI, EXTENT
6456  6136  STRNGI, STRING
6457  6140  ALPHAI, ALPHA
6460  6142  HENRYI, HENRY
6461  6144  AGAINI, AGAIN
6462  6021  TOPI, TOP
            *6500
6500  0000  0000
6501  7702  7702
6502  7762  7762
6503  3720  3720
6504  7760  7760
6505  7762  7762
6506  7742  7742
6507  3720  3720
6510  7742  7742
6511  0020  0020
6512  3020  3020
6513  7742  7742
6514  7760  7760
6515  7740  7740
6516  7740  7740
6517  3720  3720
6520  7742  7742
6521  3720  3720
6522  7742  7742
6523  2322  2322
6524  0040  0040
6525  3760  3760
6526  0744  0744
6527  7750  7750
6530  4045  4045
6531  0040  0040
6532  4064  4064
6533  7777  7777
            *6600
6600  0000  0000
6601  2110  2110
6602  3114  3114
6603  3014  3014
6604  3014  3014
6605  3114  3114
6606  2110  2110
6607  3217  3217
6610  0100  0100
6611  3774  3774
6612  1013  1013
6613  0242  0242
6614  1004  1004
```

Table 3, Continued

```
6615  4040  4040
6616  4102  4102
6617  3014  3014
6620  2110  2110
6621  3216  3216
6622  2111  2111
6623  3114  3114
6624  3770  3770
6625  1004  1004
6626  0401  0401
6627  0202  0202
6630  0101  0101
6631  5740  5740
6632  3114  3114
6633  7777  7777
            *6700
6700  0000  0000
6701  4774  4774
6702  4554  4554
6703  0504  0504
6704  0574  0574
6705  4602  4602
6706  4402  4402
6707  0444  0444
6710  4376  4376
6711  0400  0400
6712  7402  7402
6713  1202  1202
6714  0200  0200
6715  1376  1376
6716  0376  0376
6717  0574  0574
6720  4414  4414
6721  0774  0774
6722  4714  4714
6723  4544  4544
6724  0402  0402
6725  0176  0176
6726  0036  0036
6727  0376  0376
6730  2202  2202
6731  1002  1002
6732  2602  2602
6733  7777  7777
```

Button Subroutine

Figure 1 (right side) and Table 4 contain the flowchart and program listing for a subroutine which both listens for button presses, and records choices and their reaction times. The subroutine begins at location 7000_8 (3584_{10}), although it can be moved elsewhere by changing Statement 28 in the Driver program. Note that the storage location for the code of the button that was pressed (RESP) and the counter for reaction time (IRT) are assigned by the FORTRAN compiler.

This subroutine is used even when reaction time is not needed - only detection of a button press. In the latter case it will accumulate a reaction time which will be subsequently erased.

When reaction time is desired, the time is counted from the instant the subroutine is entered. Thus, with the present Driver program, time is counted from the offset of the last event displayed. It is not possible without changing this Driver program, to accumulate reaction time while the characters are still being displayed.

Table 4.

Subroutine for Button Sensing

```
                *7000
7000  0000  BUTTON, 0000          LINK BACK TO FORTRAN
7001  7200         CLA           CLEAR AC
7002  3231         DCA CNT       SET RT COUNTER TO ZERO
7003  6612         6612          CLEAR BUTTON STROBE FLAG
7004  6611  LISTEN, 6611          WAS A BUTTON PRESSED?
7005  5217         JMP MSEC      NO, GO TO ADD 1 MILLISECOND TO COUNTER
7006  6651         6651          YES, LOAD AC WITH BUTTON CODE
7007  6651         6651
7010  6651         6651
7011  6651         6651
7012  6651         6651
7013  3632         DCA I IRES    DEPOSIT BUTTON CODE IN FORTRAN LOCATION
7014  1231         TAD CNT       LOAD RT COUNTER                    FOR RESPONSE
7015  3633         DCA I IRT     DEPOSIT RT VALUE IN FORTRAN LOCATION FOR RT
7016  5600         JMP I BUTTON  EXIT
7017  2231  MSEC,  ISZ CNT       ADD 1 TO RT COUNTER; HAVE 4 SECONDS ELAPSED?
7020  5224         JMP LOOP      NO, GO TO WAIT ONE MORE MILLISECOND
7021  3632         DCA I IRES    YES, DEPOSIT 0 IN FORTRAN BUTTON LOCATION
7022  3633         DCA I IRT     DEPOSIT 0 IN FORTRAN RT LOCATION
7023  5600         JMP I BUTTON  EXIT
7024  1234  LOOP,  TAD M260      LOAD AC WITH TIME CONSTANT
7025  3235         DCA TIME      STORE IT
7026  2235         ISZ TIME      HAS 1 MILLISECOND PASSED?
7027  7026         JMP.-1        NO, CHECK AGAIN
7030  5204         JMP LISTEN    YES, RETURN TO CHECK FOR A BUTTON PRESS
7031  0000  CNT,   0000          RT COUNTER
7032  7136  IRES,  7136          FORTRAN LOCATION FOR RESPONSE CODE
7033  7137  IRT,   7137          FORTRAN LOCATION FOR REACTION TIME
7034  7300  M260,  7300          TIME CONSTANT
7035  0000  TIME,  0000          TIME CONSTANT STORE
7036  4200  BEGIN, JMS BUTTON
7037  7402         HLT           ENTRY POINT TO TEST SUBROUTINE
7040  7036         JMP.-2
```