COMPUTER TECHNOLOGY

An alphanumeric display program for a PDP-8*†

RALPH NORMAN HABER

University of Rochester, Rochester, New York 14627

Alphanumeric and special characters composed of 5 by 7 dot matrices can be displayed on a cathode ray tube by use of a program written for the PDP-8 family of computers. The CRT surface is specified to contain up to 32 rows with up to 32 characters per row. The 35 dots defining each character require 1.5 msec to generate. Every possible arrangement of spatial and temporal organization among and between strings of characters can be created by simple parameter specification before each trial. The program is designed to operate with a minimal 4k-memory computer configuration, with no special hardware other than an oscilloscope display. The program also permits efficient on-line operation so that an entire experiment can be run quickly and efficiently without further programming. A large number of examples are detailed, and operating procedures are specified.

An alphanumeric display subprogram and its FORTRAN driver program are described. Their purpose is to display strings of characters, each character composed of a 5 by 7 dot matrix on a cathode ray tube (CRT) surface. The programs have been written so that a researcher could design and run entire experiments from the console, without any further programming or hardware changes. Minimum hardware to utilize these programs is a PDP-8 family computer, with only 4k memory, plus the extended arithmetic unit. A slight program modification is needed if this option is not available. The display used is the 34D oscilloscope display system, which includes the interface control and Tektronix 503 oscilloscope. All hardware is "off the shelf" and commonly found in existing installations.

This program operates as a string generator so that letters, characters, or spaces are manipulated as predefined strings. Up to 32 strings are available on each trial, and these may be displayed on any of 32 rows defined on the CRT surface, with up to 32 characters on each row. Up to 64 different characters can be used. Since each character is 5 dots wide by 7 dots high, 35 bits are needed to define each one. These 35 bits are stored in three 12-bit words in the program and may be easily altered if special characters

are desired (see below for this procedure). The horizontal and the vertical spacing between dots within a character can be altered from trial to trial, as can the horizontal and vertical spacings between characters. See Fig. 1 for some characters.

As a character is being displayed, 35 bits are examined, bit by bit, to determine whether or not any of the 35 possible dots are to be intensified. The subroutine that performs testing and intensification of the appropriate dots requires 45 microsec to execute for each dot, whether the dot is intensified or not. Thus, the minimum duration for a single character is

1.5 msec. If the duration of a single character is to be 30 msec, then it is repeated 20 times. If three characters are to be displayed once, 4.5 msec would elapse.

The apparent duration of the characters interacts with the persistence of the phosphor on the CRT. A P2 phosphor will hold each point for 120 msec (the phosphorescence of the CRT), even though the fluorescence lasts no more than 4 microsec. Thus, if only long durations are going to be used, a phosphor of long persistence should be selected. This will permit many characters to be written without appreciable fading and consequent flickering. On the other hand, if the program is to be used to display characters for very short durations, then a short-persistence phosphor (for example, a P15 with a persistence of 20 microsec) should be selected. With a P15 only a few characters can be displayed at a time—up to about 15 or 20—without noticeable flicker. The oscilloscope CRT can be changed in a few minutes without need of service personnel. These tubes cost less than \$100 and are available from the manufacturer.

The program provides an option to

Fig. 1. Alphabetic characters derived from the 5 by 7 dot matrix. The last character lists the order in which the dots are displayed in each matrix.

*This development was supported in part by a research grant from the United States Public Health Service, MH 10753, and a grant from the National Science Foundation, GB 4547. I would like to thank Mr. Steven Barry for his generous assistance throughout the development of this program.

†The program write-up, a flow chart, and a symbolic and binary listing will be available from DECUS (Digital Equipment Corporation Users Society, Maynard, Massachusetts) after November 1, 1971.

Table 1
Control Characters to Define Parameters for the Driver Program

1 = Wait for S's start switch to initiate trial
2 = Initiate trial immediately
3 = Number of strings
4 = Length of each string
5 = Number of repetitions of each string of characters
6 = Character names (first 32 characters)
7 = Character names (second 32 characters)
8 = Location on row of leftmost character in each string
9 = Row location of each string
10 = Interstring intervals in milliseconds
11 = Number of cycles each group of strings are to be repeated
12 = Brightness level of each string (preset at maximum)
13 = Horizontal spacing between characters (preset at 1/2 width of character)
14 = Vertical spacing between characters (preset at 1/2 width of character)
15 = Horizontal spacing between dots (preset at 3 dots @ 10 V/cm)
16 = Vertical spacing between dots (preset at 3 dots @ 10 V/cm)

select one of three dot brightnesses—that is, the total time or fluorescence each dot is intensified. In addition, the manual intensity control on the oscilloscope can select the overall intensity of the CRT's electron beam. Except for the dots, the display surface is dark. If the ambient illumination in the room is average, the surface color is light green, and the color of the dots depends on the phosphor in the tube. Other CRTs have different characteristics.

The horizontal and vertical sensitivity controls on the oscilloscope determine the size of the characters by magnifying or diminishing the horizontal and vertical dimensions, respectively. The size of the dots is constant, however, at about 0.5 mm. Their luminance is independent of the horizontal and vertical control settings. If the horizontal control is set too coarsely, each character will be large. If too many characters are specified in a row, only the first few will be on the surface, the rest being off screen to the right.

The absolute location of each dot is given by two 10-bit numbers for the X and Y coordinates, respectively. Thus, the surface is treated as a 1024 by 1024 matrix. Since the D-A converters driving each coordinate provide 10 V for a full 1024 unit of deflection, setting the horizontal and vertical controls to 1 V/cm will permit the entire 10-cm surface to be addressed. When characters are written as dots, the program specifies the absolute location of the lower left-hand dot, and then increments all others from that. Each character is located with respect to the 0-0 origin in the lower left-hand corner.

This display subprogram contains 640₁₀ instructions, plus 192₁₀ locations to store the codes for the

64₁₀ characters. The driver program, written in FORTRAN II for the PDP-8 (see Haber, Barry, & Uhlman, 1970), accepts values for the parameters for each trial, for example, the names of the characters to be displayed and their durations, locations, and so forth. It also initiates each trial by sensing a start button or pacing automatically, records the S's responses, including his latencies if desired, and can contain possible routines to do on-line analyses. The description in this paper will be based upon this FORTRAN driver program. However, a number of other versions could be written, either in FORTRAN or in the same PDP-8 assembly language in which the display subprogram is written.

Before each trial, the driver program asks for the values for that trial. These replace those in the previous trial, so that any that are not changed will be repeated on the next trial. Hence, it is necessary to change only those values that differ from trial to trial.

These parameters may be entered in several ways. While the experiment is being designed or during pretesting or practice sessions, they can be typed just before each trial begins. Once the sequence of trials and their values is determined, the parameters can be punched on paper tape. Such a tape can then be read under the driver program control, trial by trial. If magnetic tape or a disk is available, these values can be read from such a bulk storage device. A brief random-number generator could be added as an additional subroutine to generate on-line trial parameters.

For each trial, up to 32 different strings of characters may be displayed. Each string must be defined as to its length, the names of the characters in it, the X and Y coordinates of its leftmost character, the number of times the string is to be repeated—which defines its duration, the horizontal and vertical spacing between the dots in the character, the horizontal and vertical spacing between each of the characters in the string, and the brightness of the string. If more than one string is to be displayed, then in addition to the above information for each of the subsequent strings, the interval in milliseconds between each string is definable. If the second string is on a different row, then a control for vertical spacing between rows is needed. If there are several sets of strings, then the number of cycles of each set is also needed. The various spacing between dots and between characters may not be altered within a string or from string to string on the same trial. They may be changed only at the beginning of each trial.

To enter trial parameters, a control character is typed first. This instructs the driver program which parameter is to be modified for the upcoming trial. These control characters are shown in Table 1. Control Character 1 shifts control to the subroutine which waits for the S to press a button to initiate the trial. This subroutine requires some minimal interface components to sense buttonpresses (see Haber, Barry, & Uhlman, 1970). Control Character 2 shifts control directly to the display subprogram so that the trial commences immediately, that is, within 0.5 msec. When the driver program is initially loaded, it inserts preset values for 12, 13, 14, 15, and 16, so that these do not have to be changed or entered unless some special condition is needed in the experiment being planned.

After typing a control character, the values of the parameters on that trial are typed in (see examples below). When all the values for a particular parameter are entered, parameter definition is terminated by typing 99. Another control character may be issued permitting another parameter to be modified. Before the first trial, it is necessary to enter each of the parameters 3 through 11. After the first trial, however, changes are needed only for those that are to be varied from trial to trial.

Character names are labeled in the subprograms as octal integers (A = 1 ... H = 10 ... Z = 32, etc.), with up to 77₈ (64₁₀) different ones permitted. Since the driver control program recodes decimal input to octal, they can be entered under driver control as decimal integers 1 to 26, etc.

Several examples of simple displays are given below. After typing the eight rows of parameters (plus spacing information if those are to be different from the preset values), typing a 2 will begin the display immediately. Typing a 1 will shift control to an external start switch. Note that the 99, which must be the last number typed for each parameter, has been omitted in these examples. See the comment on the cycle parameter just before Example 5.

(1) Display one row of six letters (e.g., the word BANANA) for 99 msec.

3. (Number of strings) 1
4. (Length of each string) 6
5. (Repetitions of each string) 11
6. (Characters) 2 1 14 1 14 1
8. (X coordinate) 1
9. (Y coordinate) 1
10. (ISI) 0
11. (Cycles) 1 0

(2) Display eight letters (e.g., RESEARCH) one at a time in the same location, each for 150 msec with 100 msec between each of them. This design treats each letter as a separate string, one character long, repeated 100 times. After a 100-msec interstring interval, the next character appears. Since the X and Y coordinates are the same for each character, they will appear on top of each other. If they are to be spaced out across the row, then the X coordinate should be changed to 1 2 3 4 5 6 7 8. To make them in a vertical column, the Y coordinates should be changed to 8 7 6 5 4 3 2 1, with the X coordinates all 1s. With suitable variation in letter position and timing, this design can easily demonstrate sequential blanking effects of adjacent letters. Further, the spatial and temporal properties of letters can be separated entirely, so that letters of a word could appear in the correct spatial position but in the wrong temporal order, etc.

3. (Number of strings) 8
4. (Length) 1 1 1 1 1 1 1 1
5. (Repetitions) 100 100 100 100 100 100 100 100
6. (Characters) 19 5 20 5 1 19 3 8
8. (X coordinates) 1 1 1 1 1 1 1 1
9. (Y coordinates) 1 1 1 1 1 1 1 1
10. (ISI) 100 100 100 100 100 100 100 100
11. (Cycles) 1 0

(3) Display a six-letter word (e.g., DESIGN) by a build-up of strings (e.g., D, DE, DES, etc.). Each string is to be on for about 100 msec, with 200 msec between each string. The different numbers of repetitions for each length may introduce slightly different levels of apparent brightness. If this is objectionable, then each string should be Length 6, with blanks (Character 0) inserted for the missing letters. Then all six repetitions should be 11. Instead of blanks, any other character could be inserted. By rearranging the order of the letters and of the X-coordinate values, any pattern of build-up or tear-down of sequences can be constructed.

3. (Number of strings) 6
4. (Length) 1 2 3 4 5 6
5. (Repetitions) 67 33 22 16 13 11
6. (Characters) 4 4 5 4 5 20 4 5 20 9 4 5 20 9 7 4 5 20 9 7 14
8. (X coordinate) 1 1 1 1 1 1
9. (Y coordinate) 1 1 1 1 1 1
10. (ISI) 200 200 200 200 200 200
11. (Cycles) 1 0

(4) Metacontrast—Display a rectangle (Character 27) for 30 msec, followed after 25 msec by two flanking rectangles for 150 msec. In this design, the second string of three characters has two rectangles with a blank inserted between them. The spacings are changed so that the adjacent rectangles have no spaces between them. This will improve the amount of metacontrast. This can be varied to achieve any degree of spatial separation or overlap desired. Example 10 illustrates another variation of this design in which the cycle parameter is used.

3. (Number of strings) 2
4. (Length) 1 3
5. (Repetitions) 20 33
6. (Characters) 27 27 0 27
8. (X coordinate) 2 1
9. (Y coordinate) 1 1
10. (ISI) 25
11. (Cycle) 1 0
13. (Horizontal character spacing) 50

This program limits a string to a single horizontal row. In order to display several rows at the same time, it is necessary to cycle parts of the trial several times. Several illustrations of this are given below. The cycling feature is also useful if the entire trial, whether of one or of several rows, is to be displayed over and over. To demonstrate apparent movement, for example, the stimuli have to be turned on and off successively with a time and space interval between them. This should be done enough times so that the S can observe and make a judgment about the nature of the perceived movement.

The cycle feature plays a critical role in the operation of the subprogram, since it sorts out the arrangements of strings and determines when the trial is over. Thus, it is necessary that the last entry in the parameter specification be a zero, signifying that after the previous cycles are completed the next one is 0. If the exit control character 99 is typed without a 0, then the program will attempt to display something 99 times.

(5) Display three rows of four letters each (A-D; E-H; I-L) for 90 msec. Each string (row) will be displayed once, requiring a total of 6 msec to do so, and then the next row is displayed, etc., until all 12 letters have been presented once. The sequence is then recycled five times for a total elapsed time of 90 msec.

3. (Number of strings) 3
4. (Length) 4 4 4
5. (Repetitions) 1 1 1
6. (Characters) 1 2 3 4 5 6 7

- 8 9 10 11 12
8. (X coordinate) 1 1 1
9. (Y coordinate) 3 2 1
10. (ISI) 0 0
11. (Cycle) 5 0

(6) Same as Example 5, but adding an arrow (Character No. 29) at the left pointing at the second row, arriving 100 msec after the array ends and lasting 200 msec. Here the three strings of four letters each are cycled five times. Then two more strings are presented—a blank, representing the interstimulus interval between the end of the first cycle and the beginning of the arrow, and then the arrow itself. It is necessary to use a blank for the ISI. If the normal ISI is used (10.0 0 100 0), then the 100-msec interval would occur on each cycle.

3. (Number of strings) 3 2
4. (Length) 4 4 4 1 1
5. (Repetitions) 1 1 1 67 133
6. (Characters) 1 2 3 4 5 6 7 8 9 10 11 12 0 29
8. (X coordinate) 2 2 2 1 1
9. (Y coordinate) 3 2 1 2 2
10. (ISI) 0 0 0 0
11. (Cycle) 5 1 0

(7) Same as Example 6 but adding a mask (filled 5 by 7 rectangle—Character No. 27) covering the second column of the letters. The mask lasts 200 msec and begins 100 msec after the arrow ends.

3. (Number of strings) 3 2 3
4. (Length) 4 4 4 1 1 1 1
5. (Repetitions) 1 1 1 67 133 1 1 1
6. (Characters) 1 2 3 4 5 6 7 8 9 10 11 12 0 29 27 27
8. (X coordinate) 2 2 2 1 1 3 3 3
9. (Y coordinate) 3 2 1 2 2 3 2 1
10. (ISI) 0 0 0 0 100 0 0
11. (Cycle) 5 1 44 0

(8) Apparent movement—Display each of two dots (Character 30) back and forth 20 times, eight character spaces apart, each for 30 msec with 80 msec between them. An ISI is placed after each dot so that an interval will occur after each. Otherwise the 80-msec delay would occur only between Dots 1 and 2 but not between Dots 2 and 1.

3. (Number of strings) 2
4. (Length) 1 1
5. (Repetitions) 20 20
6. (Characters) 30 30
8. (X coordinate) 1 8
9. (Y coordinate) 1 1
10. (ISI) 80 80
11. (Cycle) 20 0

3. (Number of strings) 16
4. (Length) 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1
5. (Repetitions) 20 20 20 20
20 20 20 20 20 20 20 20
20 20 20 20
6. (Characters) 27 27 27 27
27 27 27 27 27 27 27 27
27 27 27 27
8. (X coordinates) 1 2 3 4 5 6
7 8 9 8 7 6 5 4 3 2
9. (Y coordinates) 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1
10. (ISI) 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0
11. (Cycle) 20 0
13. (Horizontal character
spacing) 50

Several other examples are listed below to provide evidence of the flexibility of the program.

(10) Metacontrast—Same as Example 4, except that the flanking rectangles are above and below as well as beside the target. Notice that, unlike Example 4, no blank characters are used here. Number 4 could be rewritten in this form, using the cycling parameter. The duration of the center rectangle is determined by the repetition. The durations of the masking rectangles, however, are determined by the cycling, since they are made up of several separate strings.

The vertical spacing (Control 14) is also adjusted to zero.

3. (Number of strings) 1 4
4. (Length) 1 1 1 1 1
5. (Repetitions) 10 1 1 1 1
6. (Characters) 27 27 27 27 27
8. (X coordinates) 2 2 1 3 2
9. (Y coordinates) 2 3 2 2 1
10. (ISI) 25 1 1 1 1
11. (Cycle) 1 30 0
13. (Horizontal character spacing) 50
14. (Vertical spacing) 70

(11) Multiple metacontrast—Same as Example 10, except that after the four masking rectangles appear, a larger set of eight masks is presented to flank the four. They are delayed 20 msec after the four masks end and have a duration of 300 msec. It is possible to enlarge this design, adding another ring of (16) masks. However, with a short-persistence phosphor, those 16 might show some slight flicker.

3. (Number of strings) 1 4 1 8
4. (Length) 1 1 1 1 1 1 1 1 1
1 1 1 1 1
5. (Repetitions) 13 1 1 1 1 13
1 1 1 1 1 1 1
6. (Characters) 27 27 27 27 27
27 0 27 27 27 27 27 27 27 27
27
8. (X coordinates) 3 3 2 4 3 3
3 2 4 1 5 2 4 3
9. (Y coordinates) 3 4 3 3 2 3
5 4 4 3 3 2 2 1
10. (ISI) 20 1 1 1 1 1 1 1 1 1
1 1
11. (Cycle) 1 7 1 3 0
13. (Horizontal character
spacing) 50
14. (Vertical spacing) 70

(12) Classification experiment—Present two letters side by side, each on for 500 msec, but with their onsets out of phase by 250 msec. Almost unlimited spatial and temporal variations of this design can be created. If upper- and lowercase comparisons are desired, a set of lowercase alphabetic characters can be specified.

3. 3
4. 1 2 1
5. 167 83 167
6. 1 1 2 2 0
8. 1 1 2
9. 1 1 1
10. 0 0
11. 1 0

(13) Display a fixation point in Position 8 for 1,000 msec. Then move it to Position 1. Display four letters (e.g., ABCD) 100 msec after the point

moves, followed by a rectangular mask over the third letter position 100 msec after the letters end. The letters are on for 100 msec and the mask for 150 msec. The fixation point remains on at all times, and for 1,000 msec after this sequence of events. This procedure is of interest to determine whether or not the mask will obliterate the third letter position, even though the eye has moved so that the mask actually falls over the second letter in the array (designed by Michael Davidson, personal communication). Davidson has already modified the display subroutine so that the mask is triggered not by an ISI (as above) but by the output of an electrooculograph specifying the exact position of the eye.

3. (Number of strings) 6
4. (Length) 1 1 6 1 5 1
5. (Repetitions) 667 67 10 67
20 667
6. (Characters) 30 30 30 0 1
2 3 4 30 30 000 27 30
8. (X coordinates) 8 1 1 1 1 1
9. (Y coordinates) 1 1 1 1 1 1
10. (ISI) 0 0 0 0 0
11. (Cycle) 1 0

(14) Display a single letter (e.g., A) for 1 sec in Position 1. Then, after 500 msec, display five letters (e.g. TRAIN), beginning in Position 3, for 150 msec. Following a 100-msec delay, display a mask over all five letter positions for 300 msec. A subroutine can be added for this design to listen for and record the S's reaction time to ascertain whether or not the initial target was contained in the subsequent array. Slight variations in this design would permit a large number of search and memory probe experiments to be run.

3. (Number of strings) 3 0
 4. (Length) 1 5 5
 5. (Repetitions) 667 20 40
 6. (Characters) 1 20 18 1 9
 14 27 27 27 27 27
 8. (X coordinates) 1 3 3
 9. (Y coordinates) 1 1 1
 10. (ISI) 500 100
 11. (Cycle) 1 0

(15) Simultaneity of meaningful patterns—Display three letters (e.g., BAT) each for 15 msec with 5 msec between each one, spacing the letters out in a row. The S's task is to decide whether the letters were displayed simultaneously or not. Geometric forms could be constructed from line segments by designing special characters.

3. (Number of strings) 3
4. (Length) 1 1 1
5. (Repetitions) 10 10 10

6. (Characters) 2 1 20
8. (X coordinates) 1 2 3
9. (Y coordinates) 1 1 1
10. (ISI) 5 5
11. (Cycle) 1 0

(16) Perceptual growth of words through repetition—Present a word (e.g., BANANA) for 18 msec. After 500 msec, re-present it, recycling this 10 times.

3. (Number of strings) 1
4. (Length) 6
5. (Repetitions) 2
6. (Characters) 2 1 14 1 14 1
8. (X coordinates) 1
9. (Y coordinates) 1
10. (ISI) 500
11. (Cycle) 10 0

(17) Visual scanning—Present a fixation point for 1,000 msec in Position 1. Have it briefly blink off for 50 msec and then return for 500 msec followed by a 45-msec presentation of an array of 10 characters, after which the fixation point returns for 1,000 msec and then goes off. The small temporal gap in the fixation point is a signal to the S that the array will come 500 msec later. The gap can be adjusted so that it is not noticeable unless the S is directly fixating. This is a scanning design, in which S has to detect the presence of a specific character in the array, which can be varied as to probability, location,

sequential arrangement, presence of blank spaces, etc. Fixation is controlled by the fixation point. Direction of scan can be controlled by always asking S to first report the leftmost character, and then the presence or absence of the control target. To get a left fixation but a right-to-left scan, have the rightmost letter reported first, etc. Note also that the horizontal character spacing is reduced from 100 to 90. With 10 characters plus fixation point, each 50 units wide, plus 50 intercharacter spaces, the 1,024 spaces on the X axis will be exceeded and the last character lost.

3. (Number of strings) 4
4. (Length) 1 1 10 1
5. (Repetitions) 667 333 3 667
6. (Characters) 30 30 1 2 14 14 8 8 14 14 14 8 30
8. (X coordinates) 1 1 2 1
9. (Y coordinates) 1 1 1 1
10. (ISI) 50 0 0
11. (Cycle) 1 0
13. (Horizontal character spacing) 90

To define special characters, the following procedures are followed. At the bottom of Fig. 1, the 35 dot positions are numbered. These are treated as three 12-bit words. A dot to be intensified is specified by a 1—a dot to be left blank is a 0. (The last bit of

the third word is ignored.) The subroutine has three sets of 77_8 (64_{10}) locations set aside to store these constants for characters. In the present version, these occupy Locations 6500-6577, 6600-6677, and 6700-6777. Thus, the three words defining a Z (Character 32_8) are 4064, 3114, and 2602. These are stored in Locations 6532, 6632, and 6732, respectively. If a new character is to be defined, its three 12-bit words have to be determined, it has to be assigned an octal number, and then stored in the appropriate locations.

In summary, this paper has described an alphanumeric display program to operate a CRT display driven by a PDP-8 computer. While there may be applications requiring special configurations of colors, luminances, or contrasts that this program could not handle, the majority of experiments in visual information processing require only reliable stability of these factors rather than their trial-to-trial variation. Thus, with the minimal computer configuration required here, this display program puts the E's initial idea or speculation only a few minutes away from a formal experimental design ready to collect data.

REFERENCE

- HABER, R. N., BARRY, S., & UHLMAN, T. On-line FORTRAN for the PDP-8. *Behavior Research Methods & Instrumentation*, 1970, 2, 248-252.