# United States Patent [19]

**Lampson**

[11] **3,911,420**

[45] **Oct. 7, 1975**

[54] **DISPLAY SYSTEM INCLUDING A HIGH RESOLUTION CHARACTER GENERATOR**

[75] Inventor: **Butler W. Lampson,** Portola Valley, Calif.

[73] Assignee: **Xerox Corporation,** Stamford, Conn.

[22] Filed: **Nov. 23, 1973**

[21] Appl. No.: **418,509**

[52] **U.S. Cl.** .......... **340/324 AD;** 178/DIG. 6; 355/5
[51] **Int. Cl.$^2$** .......... **G06F 3/14**
[58] **Field of Search** .......... 340/324 A, 324 AD; 178/DIG. 6; 355/5

[56] **References Cited**

UNITED STATES PATENTS

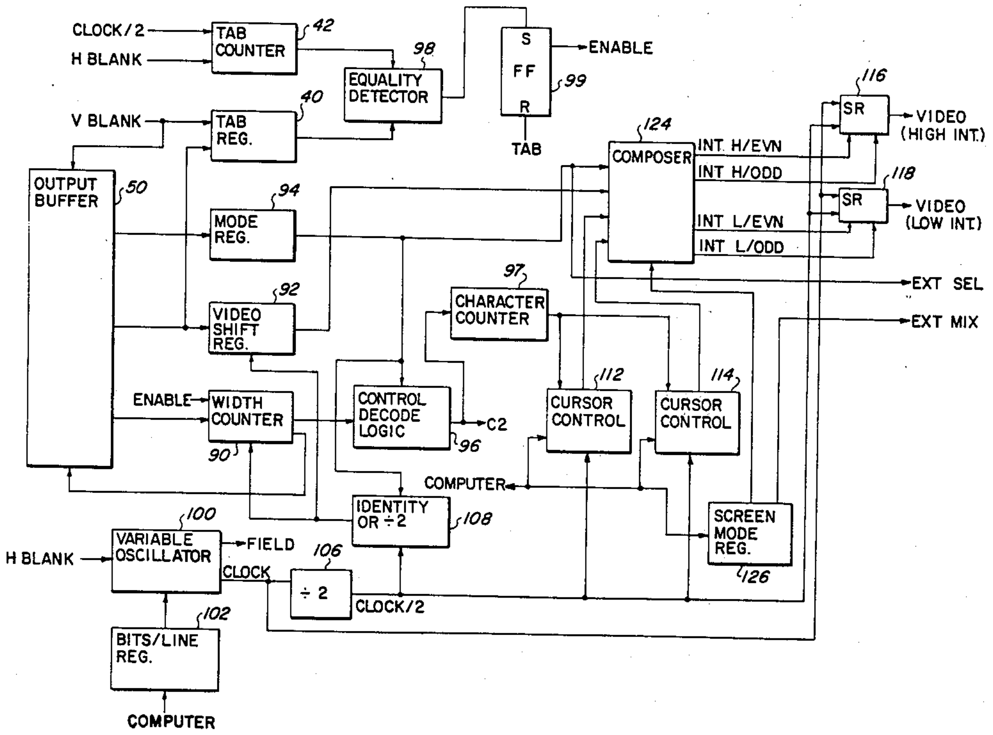
| | | | |
|---|---|---|---|
| 3,387,084 | 6/1968 | Hine et al. | 340/324 A |
| 3,426,344 | 2/1969 | Clark | 340/324 AD |
| 3,514,616 | 5/1970 | Kolb | 340/324 AD |

*Primary Examiner*—Marshall M. Curtis
*Attorney, Agent, or Firm*—James J. Ralabate; Terry J. Anderson; John H. Chapman

[57] **ABSTRACT**

The present invention relates to the display of video information on an alpha-numeric display medium by means of a character generator which is organized of random access memories and registers. The character generator organization provides for a high definition of characters, variable line width, proportional space characters, and a large number of different characters in a given font.

**24 Claims, 7 Drawing Figures**



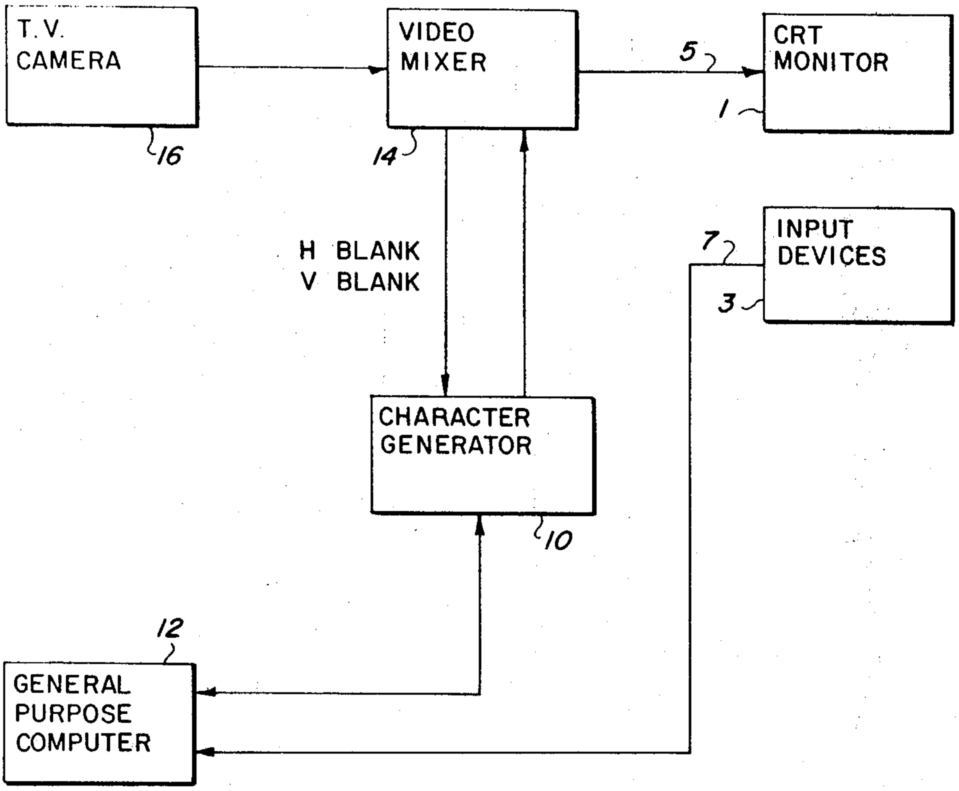

T.V. CAMERA
16
VIDEO MIXER
14
5
CRT MONITOR
1
H BLANK
V BLANK
7
INPUT DEVICES
3
CHARACTER GENERATOR
10
12
GENERAL PURPOSE COMPUTER

FIG. 1

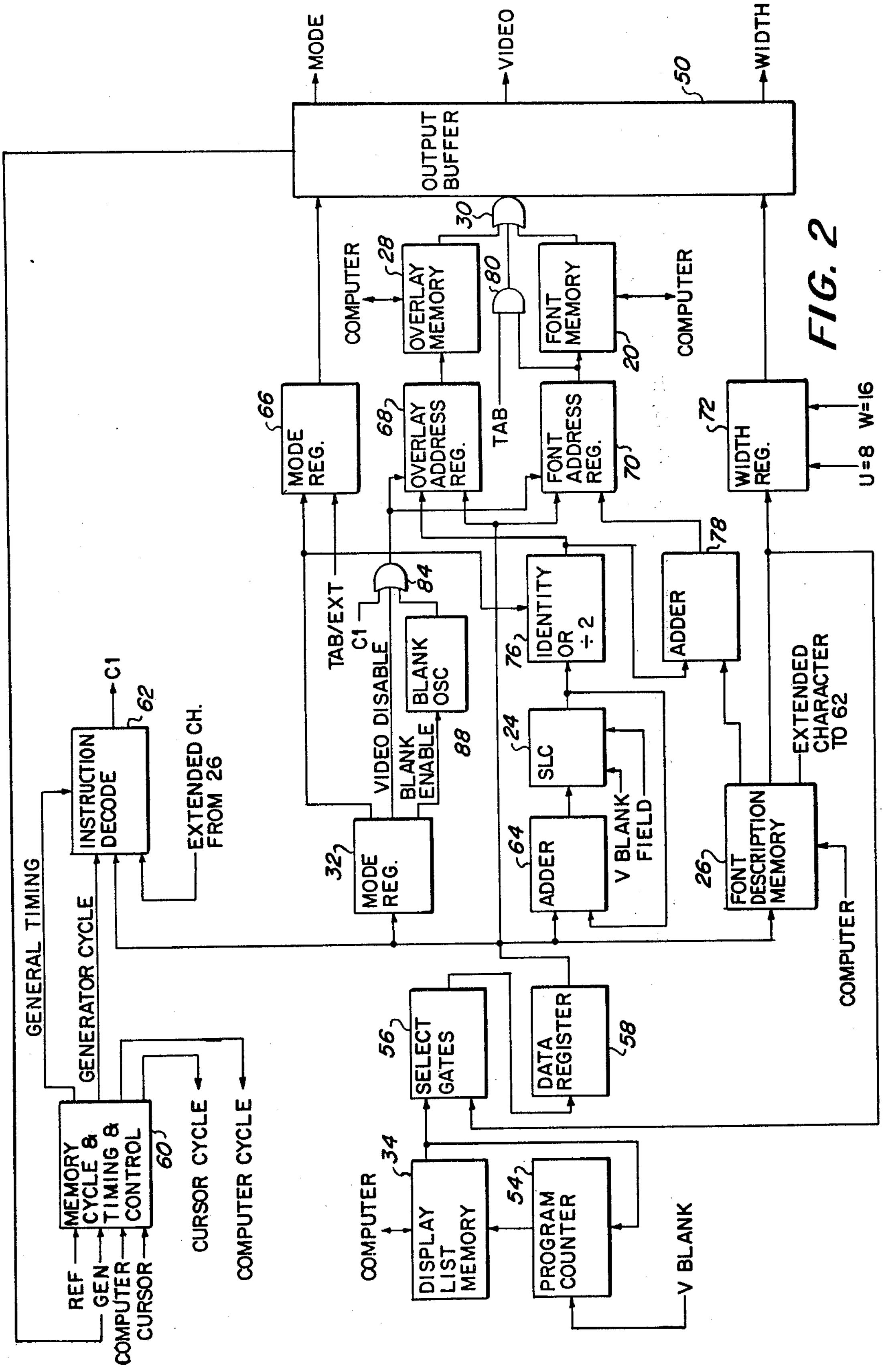

MODE
VIDEO
WIDTH
50
OUTPUT BUFFER
30
80
28
OVERLAY MEMORY
COMPUTER
FONT MEMORY
20
COMPUTER
66
MODE REG.
68
OVERLAY ADDRESS REG.
TAB
FONT ADDRESS REG.
70
72
WIDTH REG.
U=8 W=16
TAB/EXT
C1
84
VIDEO DISABLE
BLANK OSC
76
IDENTITY OR ÷2
ADDER
78
C1
62
INSTRUCTION DECODE
EXTENDED CH. FROM 26
BLANK ENABLE
88
24
SLC
EXTENDED CHARACTER TO 62
GENERAL TIMING
GENERATOR CYCLE
32
MODE REG.
64
ADDER
V BLANK FIELD
26
FONT DESCRIPTION MEMORY
COMPUTER
56
SELECT GATES
DATA REGISTER
58
MEMORY CYCLE & TIMING & CONTROL
60
CURSOR CYCLE
COMPUTER CYCLE
REF
GEN
COMPUTER
CURSOR
34
COMPUTER
DISPLAY LIST MEMORY
54
PROGRAM COUNTER
V BLANK

FIG. 2

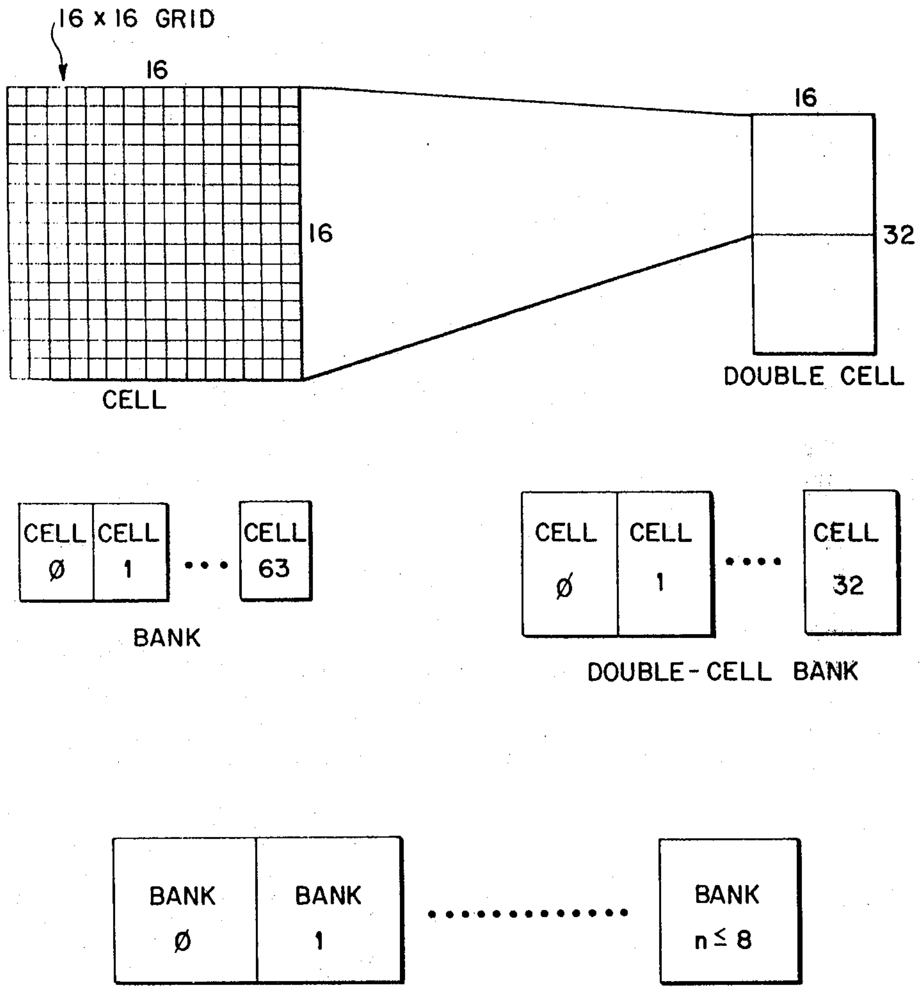


FIG. 3

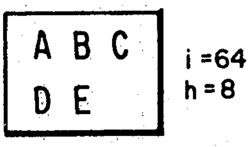


FIG. 4a

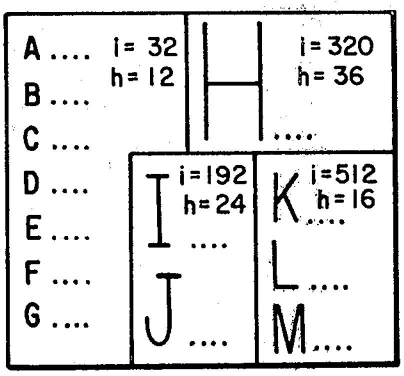


FIG. 4b

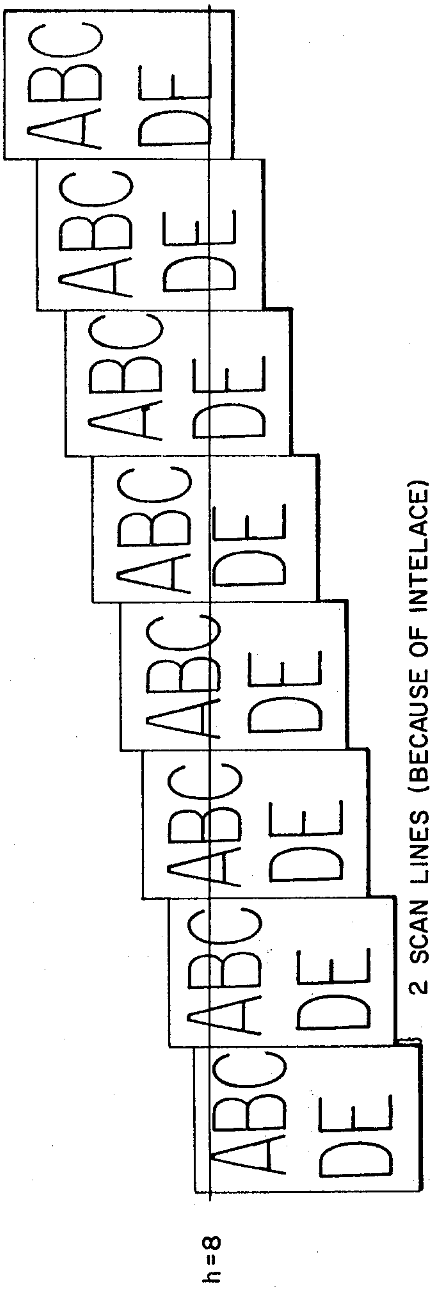

ABC
DE
ABC
DE
ABC
DE
ABC
DE
ABC
DE
ABC
DE
ABC
DE
ABC
DE
2 SCAN LINES (BECAUSE OF INTELACE)
h=8

FIG. 5

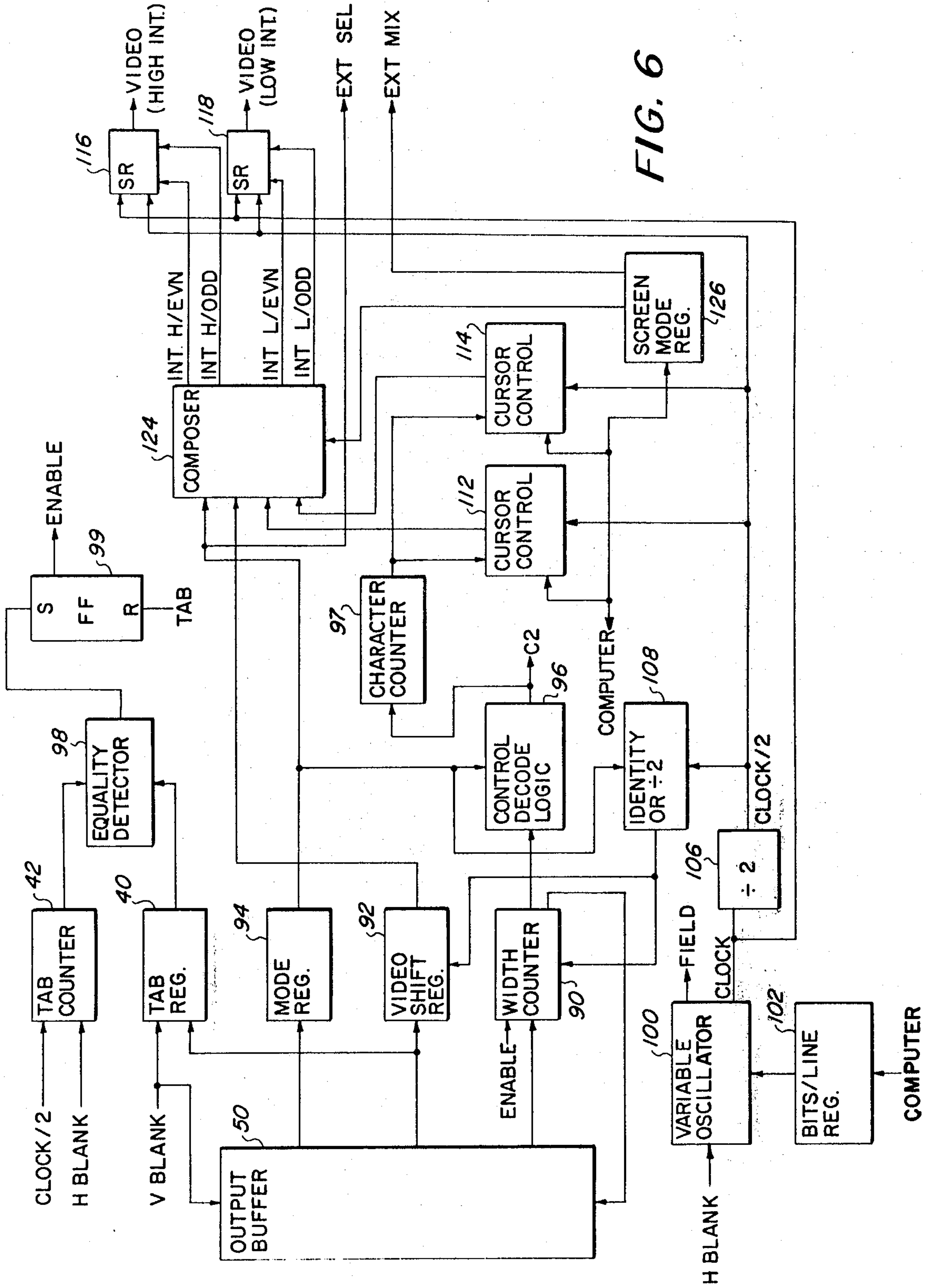

CLOCK/2
H BLANK
TAB COUNTER
42
V BLANK
TAB REG.
40
EQUALITY DETECTOR
98
S
FF
R
99
TAB
ENABLE
OUTPUT BUFFER
50
MODE REG.
94
VIDEO SHIFT REG.
92
ENABLE
WIDTH COUNTER
90
CONTROL DECODE LOGIC
96
C2
CHARACTER COUNTER
97
COMPOSER
124
INT. H/EVN
INT. H/ODD
INT L/EVN
INT. L/ODD
SR
116
VIDEO (HIGH INT.)
SR
118
VIDEO (LOW INT.)
EXT SEL
EXT MIX
CURSOR CONTROL
112
CURSOR CONTROL
114
SCREEN MODE REG.
126
COMPUTER
IDENTITY OR ÷ 2
108
CLOCK/2
÷ 2
106
FIELD
CLOCK
VARIABLE OSCILLATOR
100
H BLANK
BITS/LINE REG.
102
COMPUTER

FIG. 6

# DISPLAY SYSTEM INCLUDING A HIGH RESOLUTION CHARACTER GENERATOR

## BACKGROUND OF THE INVENTION

This invention relates to a device for generating video signals from binary information, and more particularly to a device for processing symbol information stored in digital form in a random access memory configuration for use on a display medium.

A fundamental operation in display systems is the conversion of data from its original form into information that is compatible with visual presentation. The input data may either be digital or analog, which may also include data entered into the system by means of an input device such as a light pen. The total process is generally designated by the generation functional term data conversion. The output information from digital computers, for example, is often stored in a memory device and read out from such a device onto a cathode ray tube display. Prior art cathode ray tube display devices for this purpose are generally specially constructed units utilizing relatively slow speed scanning in which the scanning beam is deflected or bent to form the symbols to be displayed in accordance with the memory output. The output information of a computer as handled in the prior art for video display, however, is not suitable for display on the screen of an ordinary television receiver in view of the relatively highspeed linear scan utilized in television apparatus.

A device taught by Johnson in U.S. Pat. No. 3,528,068 provides means for processing the output of the digital computer so that it is converted to a form suitable for display on the screen of an ordinary television receiver. His device accomplishes this result by first storing the symbol information to be displayed in a highspeed random access memory, with the information being in binary coded form. The binary coded information is read sequentially out of the memory into a symbol generator where it is translated into a series of linear dot patterns. A predetermined number of lines or such dot patterns represent the symbols to be displayed. The symbol generator is synchronized with the television cathode ray tube scan so that the dot pattern output which is fed to the video circuits of the receiver appear on the cathode ray tube in appropriate positions on the scanning raster. The symbol generator forms the dot patterns of each line of the symbols in a row in sequence, appropriate gating circuitry being utilized in conjunction with the magnetic read-out core to display the proper dot patterns at the appropriate times.

The present invention provides an organization of random access memories and control elements which provide for a high-resolution display and a combination of features such as variable line width, proportional space characters, and segmented display rasters which is not taught by the prior art.

Another object of the present invention is to provide for the generation of a large number of different characters in a given font.

Other objects of the invention will be evident from the description hereinafter presented.

## SUMMARY OF THE INVENTION

The invention provides a device for processing symbol information stored in binary coded form such that video signals are generated which may be utilized on a display medium. Specifically, the present invention provides the generation of alpha-numeric characters from the conversion of binary data by means of random access memories, registers, and control elements. A feature of the invention is that a character may be represented within memory cells defining a font memory, such cells capable of forming variant sized matrices which define the given characters. The font memory may also include an overlay memory which allows any one of a field of characters to be overlaid on any character being displayed from the font memory. Another feature of the invention is that any text to be displayed is actually stored in an additional random access memory in the form of instructions for controlling the generation of binary information to be processed. In the preferred embodiment, a computer is used to generate such binary information. The computer executes the instructions stored in this memory and generates a string of bits (binary digits) in accordance with these instructions which is used to produce video signals for a display medium.

Still another feature of the invention is that the memory and control organization of the character or video generation apparatus provides for complex rasters on a display medium. In addition to variant sized characters, a raster may be generated which has a plurality of display fields, each of which may contain different alpha-numeric representations.

These and other features which are considered to be characteristic of this invention are set forth with particularly in the appended claims. The invention itself, however, as well as additional objects and advantages thereof, will best be understood from the following description when considered in conjunction with the accompanying drawings.

## BRIEF DESCRIPTION OF THE DRAWINGS

FIG. **1** is a functional block diagram illustrating the basic elements of the system of this invention,

FIG. **2** is a functional block diagram of the display list processor portion of the character generator of FIG. **1,**

FIG. **3** is an illustration of the organization of the font memory shown in FIG. **1,**

FIG. **4** *a* is a graphical illustration of a simple display raster in contrast to FIG. **4** *b* which is a display raster constituting a plurality of display fields or segments,

FIG. **5** is a graphical illustration of the functional development of a raster, and

FIG. **6** is a functional block diagram of the video processing elements of the character generator shown in FIG. **1.**

## DESCRIPTION OF THE PREFERRED EMBODIMENT

In FIG **1** is shown the basic elements of the system which converts binary information to a video signal which may be utilized on a display medium. Display media contemplated would include, but not be limited to, television receivers, cathode ray tube display terminals, and electrostatic and graphics printers. In this preferred embodiment, however, it will be assumed that the display medium is a cathode ray tube monitor **1.** Any conventional T.V. type CRT terminal which sequentially scans the CRT screen would suffice. For optimum design, the terminal would use a 15-inch, 1029-line monitor oriented vertically in order to produce a video raster consisting of 1209-line horizontal video comprising a display area slightly larger than a standard sheet of 8½ × 11 paper. The display may further be equipped with an independent keyboard, a keyset and an input device **3**, such as a digital pointer, for positioning a cursor on the display area. A single coaxial cable **5** for the video signals and three twisted pairs **7** for digital data, i.e., input, output and clock, would connect the terminal to a central site where the character generator **10** and its associated computer **12** are located. If a plurality of terminals were contemplated, the connection would be radial in that each terminal would have its own set of connecting wires. The terminal could even include a collection facility through conventional logic design for accepting input data on the terminal and transmitting it to the controlling computer.

The input devices **3** are connected through the line **7** to the computer **12.** A general purpose computer suitable for this embodiment is the Data General Nova 1200. The binary output of the computer **12** is connected to the input of the character generator **10** which processes the binary information to generate output video signals. A video mixer **14** receives signals coming from a TV camera **16,** processes the synchronizing information which is a part of these signals, and generates a signal called horizontal (H) blank and vertical (V) blank which is transferred to the character generator **10** for synchronizing the video signals generated by the generator **10.**

Instead of the T.V. camera **16,** one could provide the necessary synchroning signals from any commercially available synchronizing source. The T.V. camera **16** is also used for providing an external video signal which is used for implementing the feature of selecting external video under display list control in the character generator **10.** Alternative sources of external video are tape recorders or other character generators. The video mixer **14** under control of the character generator **10** can select either the external video or video from the character generator **10.** The video signals processed by the mixer **14** are transferred over the cable **5** to the CRT monitor **1** for viewing.

Dot matrix representations of characters to be displayed on the monitor **1** are stored in a read/write font memory **20** within the generator **10,** as shown in FIG. **2.** The memory **20** is organized into cells, each cell containing 256 bits arranged in a 16 × 16 array as shown in FIG. **3.** There are 64 cells in a bank and up to **8** banks per terminal. A bank could be optionally configured with 32 double-cells which are made up of two cells concatenated vertically. The memory **20** could be any commercially available random access memory organized in this fashion and designed to have sufficient speed to handle the desired number of characters per line to be displayed on the monitor **1.**

A character is represented in the memory **20** by any number of cells concatenated horizontally. Either single or double cells may be used so that a character may be represented by a 16 × 16 dot matrix, or 32 × 16, 16 × 32, 32 × 32, 16 × 48, etc. Associated with each character are also two numbers. On is a width, which indicates the number of dots taken up by the character in a horizontal trace on the display screen. The width indication includes any trailing white space as well as the definition of the character itself. A second number associated with each character is a displacement which allows its respective dot matrix to be displaced upward on the text line of the display screen. The displacement provides a font whose total vertical height is greater than **16** to be represented with single cells, provided that no individual character is higher than 16. Additionally, an extension flag is associated with each character. If the flag is set, the width is assumed to be 16 plus the width of the extension and the width field for the character is interpreted by the character generating system of FIG. **2** as specifying another character denoted the "extension" which represents the next 16 dots. Since the extension is treated by the system like any other character, it in turn may have an extension so that characters of any width can be processed.

The dot matrices are in fact stored in the form of binary data or bits which appear on the display screen of the monitor **1** as small rectangles. The aspect ratio of these rectangles are extremely important for font design and may be controlled by conventional means within the terminal for optimized viewing of the display raster. The height of a character is fixed by the font definition as stored in the memory **20** and cannot be altered for a given font; however, the width of a character is controlled by the number of bits in the character definition (WX) and the speed with which these bits are sent to the monitor **1.**

The font memory **20** is indexed by a display list character code from data register **58** and five low-order bits of a scan line counter **24** with displacement (off-set) added. If the scan line counter **24** plus displacement is greater than 15 (or 31 for a 16 × 32 matrix), zeros are returned. The scan line counter **24** is a conventional register which keeps track of which row of the dot matrix should be displayed next by counting down after each successisve scan line has been traced. The bottom row may be arbitrarily numbered 0 and scanned last. Thus, if a text line occupies 20 scan lines (approximately 5mm on a 15-inch monitor oriented vertically), the counter **24** will successively count down the values 19, 18, ..., 1, 0. When the value becomes negative **20** is added back to the counter **24,** and the next text line is displayed.

A font description memory **26** contains information for the three font description parameters: character width, vertical displacement and horizontal extension. The memory **26** is a 256 word by 12 bit bipolar memory providing information for each of 256 font characters. The data is stored in the following format:

| C4 | C5 | C6 | C7 | C8 C9 | C10 | C11 | C12 | C13 | C14<br>C15 |
|---|---|---|---|---|---|---|---|---|---|
| : | DIS | :X: | | | | | WX | | : |

If $X = 0$, $WX$ is interpreted as character width; if $X = 1$, $WX$ is used to form the font memory address for the horizontal extension. $DIS$ is a vertical displacement for correct placement of characters.

Character Width ($X = 0$): This feature determines the actual number of bits to be displayed for a given character. The value in $WX$ is used to compute the actual width in the following way:

$$W_{actual} = WX + 4 \text{ (even values only)}$$

Although $WX$ has seven bits, only bits 11–14 are used for width; widths may range from 4 to 32

Horizontal Extension ($X = 1$): This feature allows definition of characters having a 32 × 16 or 32 × 32 font definition matrix. The extension indicates that a character is to be accessed in two (or more) font character locations: one pointed to by the current character and the other pointed to by *WX*. The displacement for both left and right halves is taken independently. The width for the left font description matrix is taken to be 16 while the width for the right font description (extension) is accessed in the same manner as any other character. Multiple extensions are possible.

Vertical Displacement: This feature allows vertical positioning of each 16 × 16 or 16 × 32 font definition matrix. The *DIS* field is used to compute the actual displacement in the following way:

$$DIS_{actual} = DIS \times 2.$$

This allows displacement to assume values from 0–14 in steps of two. This feature will allow fonts to have an effective height greater than the cell height used by the font.

Another memory **28** is connected in parallel with the font memory **20** to the input of an OR gate **30** for providing any one of eight characters to be overlaid on any font character being diplayed from the memory **20.** The dot matrix representation of an overlay character is simply ORed into that of the font character. The overelay character is selected by a three bit code from data register **58** and five low-order bits of the scan line counter **24** without displacement added. The overlay memory **28** is convenient for the use of cursors which lie on integral character positions and for underlines, overbars, accents, and other symbols. The font and overlay memories **20** and **38** are accessed under control of a display list memory **34** and the scan line counter **24.** The display list memory **34** is used to select the character to be displayed at each position on a scan line and control the value of the scan line counter **24,** as will be further described herein.

Overlay memory **28** is implemented with 512 × 16 bit bopolar memory, thus providing eight overlay characters each consisting of two 16 × 32 bit character definitions. The first character definition, referred to as the overlay character, is accessed when displaying a normal font character. The second character definition, referred to as the overlay extension, is accessed when displaying a font extension. Both mode and width information will be indentical to that of the character being overlayed.

The text to be displayed is stored in the memory **34** and is referred to as display lists. The text is stored in binary form constituting instructions for the character generator **10.** In order to create the display raster, the generator **10** executes these instructions and generates a string of bits which is used to modulate the electron beam of the cathode ray tube in the display monitor **1** as the beam scans across the display screen. For every scan line, the generator **10** executes instructions which produce the appropriate display for each character intersected by that scan line.

The display list memory **34** contains instructions which are divided into two classes of list memory words, display characters and control words. The list word is interpreted as follows:

```
C4   C5               C7     C8              C15
: O:         OVL      :            CHAR           :

C4   C5      C6       C7   C8                      C15
: l:    J:   OP  :         CHAR          :
```

Bit number **C4** through **C15** corresponds to computer words with **C15** the least significant bit.

Display Words (C4 = 0): CHAR is interpreted as an eight bit character to be displayed with the current mode, and is displayed with one of eight overlay characters selected by OVL.

Control Word (C4 = 1): There are four instructions that can be executed as a control word which are selected by the two-bit OP field. Each of these instructions can be modified by J to be a jump or non-jump instruction. All jump addresses are generated by taking the next 12-bit word, left shifting 1 and placing a zero in the least significant bit.

ADD to SLC (OP = 0): This control word will cause the contents of CHAR to be added to the scan line counter (SLC) **14.** If J is a zero (non-jump), this add may result in a positive or negative value for the counter **14,** and processing continues at the next word in the display list. If J is a one (jump), CHAR is added to the counter **14,** and the result is examined. If the result is non-negative, the result is placed in the counter **14** and the next word in the display list is used as a jump address. If the sum of CHAR and SLC is negative, the add is inhibited and processing continues at the next word plus one in the display list.

TAB (OP = 1): This control word causes CHAR to be placed in a TAB register **40,** shown in FIG. **5.** The register **40** may contain any number from 0 to 255, where each increment represents 32 bit times across the scan line. Whenever this control word is executed display of characters is stopped until the contents of a TAB counter **42** is found to be equal to the new TAB value, after which display of text resumes. The TAB counter **42** is cleared to zero by the horizontal sync signal of the CRT monitor **1.** The basic tab function is accomplished by setting TAB to the desired value across the scan line. The start of a new line, with automatic indenting is accomplished by setting TAB at the end of a line to a small value such as 0,1,2, etc. End of page processing can be achieved by setting TAB to some large value never reached during a scan time, such as 255. If J = 0, processing continues at the next word in the display list, or if J = 1, the next word is used as a jump address.

MODE (OP = 2): This control causes CHAR to be placed in the mode register **32.** The mode register **32** will effect the processing of characters which follow in the display list. The mode register **32** is interpreted as follows:

| | |
|---|---|
| C8 | not used |
| C9 | 0 = display character generator video<br>1 = display external source video |
| C10 | 1 = disable character generator video |
| C11 | 1 = select blinking option |
| C12 | 1 = select high intensity |
| C13 | 1 = vertical scale X2 |
| C14 | 1 = horizontal scale X2 |
| C15 | not used |

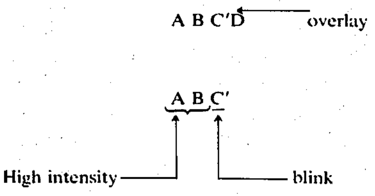
If J = 0 processing continues at the next work in the display list, or if J = 1, the next word is used as a jump address.

CONTROL (OP = 3): This control may be used for some special control functions such as halting the display processor for debug purposes, or setting flags to control some special circuits. If J = 0, processing continues at the next word in the display list, or if J = 1, the next word is used as a jump address. The following example illustrates the use of these instructions for a font having a height of 16 (20 octal).

| Address | Octal Contents | Symbolic Contents | Action |
|---|---|---|---|
| 0000 | 6020 | JI,20 | increment SLC by 20 and jump |
| 0001 | 0040 | 100 | to location 100 |
| 0002 | 6376 | JI,-2 | decrement SLC by 2 and jump |
| 0003 | 0040 | 100 | to location 100 |
| 0004 | 6020 | JI,20 | increment SLC by 20 and jump |
| 0005 | 0100 | 200 | to location 200 |
| 0006 | 6376 | JI,-2 | decrement SLC by 2 and jump |
| 0007 | 0100 | 200 | to location 200 |
| 0010 | 4777 | T,255 | tab to end display processing |
| . | | | |
| . | | | |
| 0100 | 5000 | M,O | reset mode |
| 0101 | 0040 | O,A | overlay O, char A |
| 0102 | 0041 | O,B | overlay O, char B |
| 0103 | 0442 | 1,C | overlay 1, char C |
| 0104 | 0043 | O,D | overlay O, char D |
| 0105 | 6403 | JT,3 | tab to edge +3 and jump |
| 0106 | 0001 | 2 | to location 2 |
| . | | | |
| . | | | |
| 0200 | 5010 | M,20 | set mode - high intensity |
| 0201 | 0040 | O,A | overlay O, char A |
| 0202 | 0041 | O,B | overlay O, char B |
| 0203 | 5230 | M,230 | set mode - OVL group 1, high int., blink |
| 0204 | 0442 | 1,C | overlay 1, char C |
| 0205 | 6403 | T,3 | tab to edge +3 and jump |
| 0206 | 0003 | 6 | to location 6 |

Display list processing is automatically started at address 0 with a scan line count of 0 or 1 (to provide for proper T.V. interlace) at the end of each vertical retrace line. A suitable display list is given below, with all the numbers in octal notation:

Assume the desired display is



The display list memory **30**, then, contains five operations which may be executed:

1. display character C with overlay V. The overlay is chosen from a set of eight 32 × 32 dot matrices and is simply ORed with the dot matrix for C.

2. tab to horizontal position *n*. Display of the next character starts at position *n* (which must be a multiple of 32). If *n* is less than the current position, the next character starts at position *n* on the next scan line. If *n* is very large, this instruction has the effect of 'end of frame.'

3. set the display mode, which controls the following features:

| | |
|---|---|
| intensity: | full, half or off |
| blink: | on or off |
| size: | standard, double width, double height, or double size |

4. increment the scan line counter (SLC) by +*i*. This can be used to subscript or superscript the following characters, or, in conjunction with the jump instruction, to format the screen.

5. jump to location L and perform (1), (2), (3), or (4). If the jump is combined with an increment, the combined increment-jump operation words as follows:

```
if SLC + i ≥ 0 then begin: SLC = SLC + i; go to L
  end else do nothing.
```

In FIG. 4 *a* is shown a simple raster on picture on the display screen with eight scan lines per text line and two-way interlace, with the first character of each line at position **64** in accordance with the following display list:

| location | contents | comments |
|---|---|---|
| 0 | increment 6; jump 100 | start here on new field with SLC = 0 or 1 – |
| 100 | tab 64 | start a new scan line at position 64 |
| 101 | display A | |
| 102 | display B | |
| 103 | display C | |
| 104–5 | increment -2; jump 100 | 2 is because of interlace |
| 106–7 | increment 6; jump 200 | SLC is now 0 or 1. It becomes 6 or 7 since the height of the next text line is 8. |
| 200 | tab 64 | |
| 201 | display D | |
| 202 | display E | |
| 203–4 | increment -2; jump 200 | |
| 205 | jump 300 | |
| 300 | tab 4096 | wait for end of page |

At the start of each field, control goes to the display list address **0** with SLC = 0 or 1 depending upon whether it is the first or second field of a frame. A text line has been assumed to occupy eight scan lines and hence is executed four times, as SLC takes on the value 7, 5, 3, 1 or 6, 4, 2, 0 depending on the field. Successive text lines are entries in a list whose elements are jump instructions.

In FIG. 4 *b* is shown a complex raster which has segmented fields of different characteristics; it is provided for by using a combination of jumps and tabs in accordance with the following display list:

| location | contents |
|---|---|
| O | increment 10; jump A |
| A | tab 32; display A; ...; increment 24; jump H |
| B | tab 32; display B; ...; increment 12; jump H |
| C | tab 32; display B; ...; increment 12; jump H |
| D | tab 32; display D; ...; increment 12; jump I |
| E | tab 32; display E; ...; increment 0; jump I |
| F | tab 32; display F; ...; increment 12; jump J |
| G | tab 32; display F; ...; increment 0; jump J |
| H | tab 320; display H; ...; increment -26; jump A; increment B; increment -2; jump C; increment 10; jump D |
| I | tab 192; display I; ...; increment -8; jump K; increment 8; |
| J | tab 192; display J; ...; increment -16; jump L; increment 0; |
| K | tab 512; display K; ...; increment -6; jump D; increment 6; |
| L | tab 512; display K; ...; increment -10; jump E; increment 2; |
| M | tab 512; display M; ...; increment -14; jump F; increment G; jump END |
| END | tab 4096 |

*i* is the identation for each display area, and H is the height in scan lines on the text line. The tabs allow the left margin of each display area to be set conveniently without regard for what appears to its left. The increment and jump sequence allows the computer, as it reaches the right margin of each display area, to determine which text line of the next area is intersected by the current scan line and to compute the proper value of the counter (SLC) **14.** Any combination of text lines may be handled in this manner.

The increment and jump sequence at the end of a scan line is precisely the same as the sequence between display areas. This phenomenon may be explained by the graphical representation of the picture of FIG. **5** displayed with loops which result from successful jump instructions which are functionally unwound, in such a manner that an entire field is displayed by the intersection of a single scan line with many reproductions of the pictures.

The memory within the mode register **32** may be organized as follows:

| Bits | Contents | Used by |
|---|---|---|
| 0–8 | Intensity/color | Composer |
| 9 | Background value (white/black) | Composer |
| 10 | Blink | Composer |
| 11 | Horizontal size | Composer |
| 16 | Vertical size | Font |

These bit locations thereby identify the functions indicated by the content designations and use. For example, the ninth bit would contain either a 0 or 1, respectively indicating a black background or a white one. Bit location **10** could indicate the blocking of a display of a character to provide a blinking indication on the display screen to call attention.

Information from the mode register **32** regarding intensity, blink, and horizontal size is fed to an output buffer **50** which is interposed between the output of OR gate **30** and the video output system, shown in FIG. **6,** to smooth out the timing irregularities due to varying character widths. The buffer **50** also permits the character generating elements to run during the fly-back time of the CRT scanning system shown in FIG. **1.** The buffer **50** holds the 16 bits of scan line video, the four bits of character width, and any new values of mode or tab.

The buffer **50**, as described in U.S. Pat. application No. 418,507 filed on Nov. 23, 1973 and assigned to the assignee of the present invention, provides for a 16-word entry on a first-in, first-out basis. Generally speaking, the implementation would come from a storage medium with a read pointer, a write pointer, and a fullness count by means of four-bit counters or registers. The location of the buffer **50** between the gate **30** and the video output ensures that the video signal will continue to be produced while the processor elements of the systeminputted to the buffer **50** are handling jumps, increments, mode changes or characters, which take less than basic memory cycle time to display. This achievement is provided by the particular organization and interrelationship of the processor elements of FIG. **2.**

As has already been described, a display list is assembled in the computer **12,** which list constitutes a string of instructions indicating what characters are to be displayed on the screen, at what position the characters are to be displayed, and what kind of modes are to be used. This binary information is transferred to the display list memory **34** where the processing to video information commences. Font information is also assembled and stored in the computer **1** whereupon at some point in time is transferred to the font memory **20,** the overlay memory **28,** and the font descriptor memory **26.**

Other external information is derived from the signals vertical and horizontal blank and FIELD. The signal vertical (V) blank is inputted to both the program counter **54** and the scan line counter **24.** As well, a signal FIELD, which contains T.V. field information from the horizontal (H) blank signal through an oscillator **100** shown in FIG. **6,** is inputted to the scan line counter **24.** These signals ensure that during vertical blank time, the program counter **54** is cleared to zero and the scan line counter **24** is set to zero or one, depending on the T.V. field.

At the end of vertical blank the character generating elements of FIG. **2** start processing information in the display list stored in the memory **34** starting at address zero as is indicated by the program counter **54.** The information fetched is passed through the select gates **56** to a data register **58.** The counter **54,** gates **56,** and register **58** are conventional electronic components. The counter **54** could be implemented by a 74161 TI module, whereas the gates **56** and register **58** may be implemented by a a 74298 TI module. The process of transferring the original binary information to and loading it into the data register **58** takes approximately one memory cycle.

The display list memory **34** and the font memory **20** are dynamic MOS memories. These memories have timing requirements for performing read or write memory cycles. Signals for controlling such requirements are generated from a memory cycle timing and control element **60.** Inputs to the element **60** are requests for the initiation of access to the various memories of FIG. **2.** One input is refresh which satisfies a requirement of dynamic MOS memories to preserve data in the memories by initiating a refresh cycle every two milliseconds.

Another source of a request for a memory cycle is the character generator **10** itself. This request is indicated by an output of the output buffer **50,** identified as GEN in FIG. **2.** Another request is from the computer **12.** If the computer **12** is to access one of the memories or registers, e.g., write new information in the list memory **34** or new font in the font memory **20,** the computer **12** generates a line that represents a request into the control element **60** at a priority somewhat lower than the character generator request. The last request to the element **60** is generated by cursor logic which is described hereinafter.

The requests, i.e., the refresh, generator, computer, and cursor signals, are ordererd by their priority. The highest priority request is the refresh signal. If the character generator **10** makes a request for memory access and there is no refresh request, then the character generator **10** would be given priority. If both the computer **12** and the character generator **10** request memory access, then the character generator **10** will get preference and the computer **12** will be ignored. The cursor request is assigned the lowest priority. The control element **60** generates certain control outputs: general timing and generator cycle signals go to an instruction decode element **62** which coordinates the distribution of control information to other elements within the system; computer cycle signals go to the computer **12** that indicate a memory cycle for the computer is taking place; and cursor cycle signals to the cursor logic that indicate a memory cycle is taking place for the cursor control elements **112** and **114** in FIG. **6.**

The memory cycle timing and control circuit number 60 are standard timing circuits for providing the necessary trains of timing signal pulses for transferring data into and through the generator **10.** To create the timing pulses, a plurality of one-shot multi-vibrators may be used to produce a series of successive timing pulses that are selected to provide such transfer of data. Memory request information, i.e., refresh, character generator, computer, or cursor memory cycles requests, may be processed by utilizing conventional modules which perform the functions described above.

The instruction decode element **62** is comprised of conventional decoding logic which is employed to produce an output signal C1 that is indicative of a given desired function based upon the inputs to the element **62.** For example, a number of AND gates and OR gates may be combined logically to take the binary information stored in the data register **58,** determine what kind of instruction is stored there, take that information combined with timing pulses from the control element **60** and generate the output control pulses. As an example, if bit **6** is on and bit **7** is off in the data register **58,** then a mode instruction is indicated. This instruction is decoded with an AND gate. The output of the AND gate is then inputted to another AND gate which has as a second input an end of cycle pulse coming from the control element **60.** A pulse is thereby generated which is tranferred to the mode register **32** to cause the register **32** to be loaded.

When information from location zero in the list memroy **34** is entered into the data register **58,** the program counter **54** is incremented by one under the control of the signal C1 from the decode element **62.** At this time the program counter **54** has a number 1 in it, and another memory cycle is commencing. With the start of a new memory cycle, information from address 1 of the display list memory **34** is to be processed and the data from address 0 in the data register **58** is to be further processed simultaneously by the character generating elements of the system shown in FIG. 2.

The information in register **58** is further processed upon the determination by the decode element **62** of whether it represents a character for display on the monitor screen or one of the various control type words which have been described as being contained in the display list memory **34.** For example, the information could represent a mode change word, a word to modify the scan line counter **24,** or perhaps a word to set a TAB. If the register **58** is found to contain a mode change word, then at the end of the next memory cycle the mode information contained in data resiter **58** is loaded into mode register **32.** When information is transferred from the register **32,** the data output from the list memory **34** located in address 1 is loaded into the data register **58,** the program counter **54** is again incremented and another memory cycle starts. This sequence represents a typical memory cycle.

If the information in the data register **58** had been an "add to the scan line counter," then the information in the data register **58** would have been added through adder **64** along with the current contents of the scan line counter **24.** The output of the adder **64** is then added back into the counter **24.** The output of the adder **64** represents the sum of its two binary inputs. At the end of the memory cycle, the decode element **62** will generate a control pulse C1 which is transferred to the scan line counter **24** to load a new value. The new value of the counter **24** will be the sum of its present value and contents of the data register **58.** The control signal C1 represents a connection between the decode element **62** and the information processing elements of FIG. **2.** C1 represents load and increment signals to the program counter **54** at the appropriate times. C1 also represents switching the select gates **56** to select between the output of the display list memory **34** for normal instruction or the output of font descriptor memory **26** for an extended character. It further represents control to the data register **58** to receive the information from the select gates **56** at the end of each memory cycle which requires it. It represents control to load the contents of register **58** into the mode register **32** if the data register **58** contains a mode change word, to load a new value into the scan line counter **24** at the end of a memory cycle if the data register **58** contains the appropriate instruction. Additionally, C1 represents control to load new information into a mode register **66,** overlay address register **68,** font address register **70,** and width register **72,** if the data register **58** contains a normal character to be displayed. The registers **66,** 68, 70, and **72** are loaded simultaneously when the data register **58** contains a character word.

In the situation of a normal character to be displayed, at the end of the next memory cycle the character address of the word contained in the reigister **58** is loaded into the character portion of the font address register **70.** Any overlay bits are loaded into the overlay address register **68.** Information from the counter **24** is also loaded into the overlay address and font address registers **68** and **70** at this time is required. An overlay address is a combination of the particular overlay character consisting of three bits of information and the pointer to the vertical position within the overlay character being processed.

The information from the counter **24** represents the contents of the scan line counter **24** either directly or divided by two which is a function of the element **76** under the control of the mode register **32.** The choice of identity or divide by two indicates whether or not the charater is to be scaled vertically times two or not. If it is not scaled, then an identify address is transferred. If it is scaled by two, i.e., twice its normal height, than the value of the scan line counter **24** is divided by two and is transferred into the overlay address register **68.** The functions of the element **76** may be provided by a 74157 TI module.

Control of the element **76** by the mode register **32** is provided by a select signal resulting from a binary digit (bit) in the mode register **32** indicative of the last time the mode register **32** was loaded from the data register **58** from the list memory **34.** Thus, the register **58** is actually under display list control to set a bit in the mode register **32** to vertically scale a character or not.

Similarly, the address for the font address register **70** is derived from contents of the scan line counter **24** through element **76** either directly or divided by two. In addition, the output of element **76** is applied to the input of an adder **78** having two inputs: the scan line count from the element **76** and vertical offset information as contained in the font descriptor memory **26.** The offset information consists of three bits which are used to subtract a number from the scan line count for generating a resultant output that is transferred to the font address register **70.** By subtracting a number, a character is effectively moved vertically up the screen (displaced). Therefore, a vertical offset is performed by subtracting some number assigned by the font descriptor of the memory **26.** The font descriptor memory **26** contains at this time the font description for the appropriate character since the address input to the font descriptor **26** is the character address as contained in the data register **58.**

Additional outputs of the font descriptor memory **26** are either width or extension information. The width information is both transferred to the width register **72** as width information or back through the select gates **56** to data register **58** as a new character, that is, the extension of the character being processed. The feedback from the descriptor memory **26** actually produces the extension of a character within the register **58.** A bit within the descriptor memory **26** indicates whether or not there is to be an extension, which is represented by an extended character signal to the decode element **62.**

If there is not to be an extension, then the width information transferred to the width register **72** will determine the width of the new character. Since such width information may be different from the width information relating to another character, as represented by the character code in the register **58,** variable widths may be provided for each character. If an extension is indicated, the extension information contains a new character code which serves as a new address to the memory **26** for presenting new width information which will complement the extension operation. These features thereby provide for the proportional spacing of characters on the display screen.

For special characters, if the font descriptor of memory **26** indicates an extended character is being processed, then the width register **72** will not be loaded with width information from the memory **26.** Instead, it will be loaded with a constant *w*, i.e., a value to indicate a width of **16.** If a TAB instruction is contained in the data register **58,** then another procedure is operational. For example, the width register **72** is forced to contain another constant *u*. In this preferred embodiment, the width of a TAB *u* is eight. TAB is a quasicharacter which has been previously discussed in principle and is to be processed differently than a true character.

The values *u* and *w* are derived from the implementation of the width register **72.** The width register **72** is an integrated circuit (74298 TI) which contains both four bits of memory and four bits of select gates. An input to the width register **72,** either the output of the font descriptor memory **26** or another input to the register **72** simply tied to a ground potential or left floating, is selected to indicate ones or zeros to cause a value indicative of the width u or *w* to be loaded into the width register 72.

If a TAB character is being processed, then the TAB value held in the data register **58** is loaded into the character address of the font address register **70.** At the same time, a bit in the mode register **66** is set which indicates that characters being processed in either a TAB or an extension. This bit is used in conjunction with the value in the width register **72** to direct the particular processing of a character, depending upon whether it is a TAB or an extension. A TAB character is being processed if a tab extension bit is set and a value of eight is in the width register **72.** On the other hand, an extended character is being processed if the tab extension is set and a width of **16** in the width register **72.** Thus, TAB and extensions are processed as characters while indicating by means of the tab extension bit that they are special characters.

The addresses of characters or special characters thus stored in the font and overlay address register **70** provide the accessing of the font and overlay memories **20** and **28,** respectively. The base character and the overlay character accessed in memories **20** and **28** are thereby selected for display and read out of their respective memories to their respective inputs to the OR gate **30** for providing video information to the buffer **50.**

An additional source of information to the buffer **50** is the output of the font register **70** directly gated through an AND gate **80** which is ORed along with the outputs from the memories **20** and **28** to the OR gate **30.** This third source of information through the gate **30** is only operational during the processing of a TAB character. Upon the incidence of a TAB input to the AND gate **80,** the TAB value stored in the register **70** is gated through to provide TAB information to the buffer **50.** This information is stored in the buffer **50** in lieu of any video information as, at the same time, any video output from the memories **20** and **28** is inhibited.

The address stored in the overlay address register **68** and the font address register **70,** respectively, contain a control bit to indicate that the scan line count address is invalid and that the overlay memory **28** or the font memory **20,** respectively, should return zeros. One condition of an invalid address is that the scan line count value entered into the registers **68** and **70** are too large, i.e., greater than the defined character matrix. Since overlays are always 32 scan lines high, if the scan line count value entered into the register **68** persents an address which is greater than 31, the control bit is set to indicate an invalid addrеss. If the address in the font address register **70** is greater than 31, a similar indication is made if the control bit in the register **70** is set to indicate that the font memory **20** should return zeros. In this way, invalid addresses are not allowed to be processed into video signals.

These two control bits, contained in the addresses of registers **68** and **70** respectively, perform additional functions. If the data register **58** contains a TAB, when a control signal **C1** is generated from the decode element **62** to set the control bits in both the overlay address register **68** and the font address register **70** to force zeros to be returned from the memories **28** and **20** in the next memory cycle. The signal also sets a bit in the mode register **66,** at the same time, to indicate a video disable signal which inhibits the processing of video information even though the character is defined. The video disable signal is gated through an OR gate **84,** along with the signal **C1,** and presents the invalid address bit in the registers **68** and **70,** respectively.

Mode register **32,** in this preferred embodiment, contains a bit to indicate that the character is to be blinking, and if such is indicated by a blink enable signal ORed with the video disable and **C1** signals, the bit enables a blink oscillator **88** to alternatively disable or not the control bits in registers **68** and **70,** depending upon whether the blink oscillator **88** is on or off. The oscillator **88** may be a one-shot multivibrator such as a Fair child 9601 device. Therefore, any one of these three signals, i.e., **C1,** video disable, and blink enable, can cause the output of the OR gate **84** to be high to set the control bits in the overlay address register **68** and the font address register **70** to disable the respective outputs from the overlay memory **28** and the font memory **20** during the next memory cycle.

At the same time, the registers **66, 68, 70,** and **72** are loaded the next character is being processed. The new information relating to it stored in the data register **58** is examined by the decode element **62** to advance its processing through another cycle to storage in the registers **66, 68, 70** and **72.** At the same time the registers **66, 68, 70** and **72** are being loaded with new information, the output buffer **50** is being loaded with the previous character, i.e., the contents of mode register **66** is loaded into the output buffer **50;** the output of any video or TAB information, whichever is gated through OR gate **30,** is loaded into the output buffer **50;** and the contents of the width register **72** is loaded into the output buffer **50.**

Thus, for a character to be completely processed, a display list memory cycle, a data register examining cycle, a font memory access cycle are necessary. While processing a given character involves three memory cycles, a new character is processed every memory cycle because the system elements in FIG. **2** are operating independently and simultaneously of one another. This processing of characters herein described gives an extremely high throughput and yet allows for complex processing necessary for a very high resolution character display.

In FIG. **6,** is shown the video processor portion of the character generator **10.** The processing elements of FIG. **5** process width, video, and mode information which is read on a first-in — first-out basis out of the buffer **50.** Width information is loaded into a width counter **90,** video information into video shift register **92,** and mode information goes into a mode register **94.** The mode information corresponds to information which was originally derived from the mode register **32** and processed through to the output buffer **50.** The information stored in the width counter **90** which defines a value or state used to control the operation of a control decode logic circuit **96.** The value in the width counter **90** is fed back into the output buffer **50** to control the timing of reading and writing of this buffer. When the state of the width counter **90** goes below a certain value, e.g., four, counter **90** makes a new request for information from the output buffer **50.** When the counter value goes to zero, then the new information which is available on the output of the buffer **50** goes into the counter **90,** shift register **92,** and mode register **94.**

When a character is read from the output buffer **50,** its associated video information is actually loaded into two shift registers constituting the register **92.** With 16 bits of video, two eight bit long shift registers are utilized. Beginning with the first bit, every odd bit is stored in one of the shift registers and the alternate, or even bits, are stored in the other shift register. The two shift registers are operated in parallel to process the odd and even bits simultaneously.

The control circuit **96** controls whether the video output information from the buffer **50** is loaded into the shift register **92** or, alternatively, the tab register **40.** When the width count of the counter **90** goes to zero, the circuit **96** determines from this condition and the value stored in the mode resister **94** whether the next character to be read from the output buffer **50** is an actual character, the extension of a character, or a tab character. If it is an actual character for display, when the circuit **96** generates a control pulse C2 to load the video shift register **92** with the video output of the buffer **50.** If the next character is a tab character, a different C2 pulse is generated to load the tab register **40** with the video output information. If the character is an extension, then the pulse C2 is still connected to load the video shift register **92.**

When a pulse C2 is generated for the first two control functions, it is also important to a character counter **92** where a count of characters is maintained as characters are loaded in the shift register and cleared when the tab register **40** is loaded. In the case of a character extension, the pulse C2 is indicated from going to the counter **120.** Therefore, the counter **97** keeps track of the modes of characters that have been processed since the last tab character.

When control circuit **96** is comprised of conventional decoding logic which is employed to produce an output signal C2 that is indicative of the above desired functiona based upon the input signals to the circuit **96.** For example, a number of AND gates and OR gates may be combined logically to take the input signals to the circuit **96** to generate the appropriate C2 signals. The width counter **90** may be implemented by a module 74161 TI which includes a chip having an overflow output that indicates a width of O. Thus, when the counter **90** goes to zero, the overflow signal is ANDed within the circuit **94** with the tab extension bit, inputted to the circuit **96** from the mode register **94,** to determine whether the character information being read out of the buffer **50** is a TAB, an extension, or a simple character. Upon the incidence of the clock/two pulses, the appropriate C2 pulses are generated from the decode circuit **96.**

The information from the output buffer **50** is processed differently if it represents a TAB. The TAB extension bit stored in the mode register **94** signals the control logic element **96** that TAB information is being read out of the buffer **50.** The control signal C2 generated from the element **96** will inhibit the loading of information into the shift register **92** leaving it empty to shift out blank video signals. The information otherwise loaded into the register **92** is loaded into the tab register **40** as the new TAB value, and the flip-flop **99** is reset to zero thereby turning off the enable signal, which is fed back to the width counter **90,** stopping the width counter **90** from functioning. As long as the flip-flop **99** is reset, the width counter **90** does not count, and the output buffer **50** will not be accessed for new information. Since the loading of the shift register **92** depends on the output of the counter **90,** the shift register is forced to shift only zeros upon this conditions, preventing a new character from being displayed on the CRT monitor **1** until a predetermined point on the screen is reached.

An equality detector **98,** a conventional comparator circuit, compres the value of the tab counter **42** with the value of the tab register **40** to determine whether or not these values are equal. When the two registers **40** and **42** contain the same value, the flip-flop **99** is set to enable the width counter **90.** The tab counter **42** has as inputs a bit clock/two signal and synchronizing signal, horizontal blank. The counter **42** increments on the clock/two signal and gets cleared to zero with the horizontal blank signal.

The tab function is thus implemented. Briefly stated, when a tab value is loaded from the buffer **50**, processing of characters is halted until the state of the tab counter **42** is incremented to the same value as the new state of the tab register **40**. When this equality takes place, the processing of characters is initiated. The usual tab function, in this preferred embodiment, is to direct the information or characters with respect to predetermined or selected points (tab value) on the monitor screen. This function could be termed tabbing to some point to the monitor screen.

The tab function may even be used to start the display of information on a new line by loading the tab register **40** with a small value such that an equality is not reached. Even though the tab counter **42** continues to be incremented, the H blank signal occurs first clearing the counter **42** thereby setting it to zero. Then, the tab counter **42** starts incrementing again to reach an equality, depending upon the value in the tab register **40**. When the equality is reached and processing begins, the video output will be displayed at the beginning of the next scan line.

The tab function may also be used to stop processing for the entire monitor screen by placing a large value, such as **255**, in the tab register **40**. The tab counter **42** is thereby always cleared to zero by the H signal and will never reach the value in the register **40**. Processing does not take place because the enable flip-flop **99** is always reset during this condition. Processing of a new page may be achieved by inputting a vertical blank signal into the tab register **40** and clearing it to zero. Then, processing will start on the next horizontal blank signal which clears the tab counter **42** to continue the processing of characters.

The width counter **90** is decremented and the video shift register **92** is shifted in accordance with a clock output from a variable oscillator **100**. The regulator **92** is always shifted in accordance with this pulse train; whereas the counter **90** is only decremented when it is enabled by setting the flip-flop **99**. The oscillator **100** may be implemented by a conventional oscillator, although one especially suitable for this preferred embodiment is described in U.S. Pat. application Ser. No. 418,507 filed Nov. 23, 1973, and assigned to the assignee of the present invention.

The character generator **10** contains a variable oscillator **100** for a bit clock. The bit clock signal provides the timing for shifting out new video information in serial stream for display on each scan line of the monitor screen. The variable oscillator **100** is loaded with a value from a bits/line register **102**. This value represents the number of bits that is desired for each scan line and is stored in the register **102** under the control of the computer **12**. The oscillator **100** also has as an input the horizontal blank signal for synchronization and is set to whatever frequency determines the correct number of bits to be shifted out for each scan line, thus providing the desired aspect ratio for the characters to be displayed. The output of the oscillator **100** labeled clock is fed directly to a divide by two element **106** which provides a clock/two signal. The clock/two signal is processed through a scaling element **108** and used as a signal to control various processing elements shown in FIG. **6**, including counting in the width counter **90** and shifting signals out of the video shift register **92**.

The scaling element **108** provides horizontal scaling for the character being processed if the display list has indicated that it is to be provided during processing. A bit from the mode register **94** is inputted to the element **108** to allow only every other clock pulse of the clock/two signal to be passed to the counter **90** and the register **92**. Passing only every other clock pulse will have the effect of causing the width counter **90** to run at half speed thereby causing bits to be shifted out at half speed. Half speed processing produces characters which are twice as wide on the screen. Therefore, horizontal scaling as provided by the element **108** doubles the width of the character. If no control bit is received from the mode register **94**, scaling does not take place and the clock/two signal is passed through as an identity. The element **108** is implemented as described in U.S. Pat. application Ser. No. 418,508, filed on Nov. 23, 1973 and assigned to the assignee of the present invention.

In addition, clock/two signal goes into cursor control circuits **112** and **114** for the horizontal positioning of the respective cursor which they control. This signal also goes into output shift registers **116** and **118** to control the shifting or loading of these registers.

A composer **124** receives the odd and even video signals generated in parallel from the video shift register **92** and further processes them through to the output register **116** and **118**. Another input to the composer **124** is the associated mode information, i.e., high (H) and low (L) intensity signals, from the mode register **94**. Still other inputs are from the cursor control circuits **112** and **114**, which provide on and off control for the cursor video and intensity signals. Yet another input to the composer **124** is a background signal from a screen mode register **126**.

The mode register **126** is loaded from the computer **12** to store three bits of information. One of them is the background information which determines whether black or white video is to serve as the display background. This background information is fed to the composer **126**. Another bit indicates external mix. When the external mix signal is fed out to the mixer **14**, if external video is selected, this bit determines whether external video only or an added mix of the output of the character generator **10** and external video will be displayed on the monitor **1**. The third bit indicates an enable to the character generator **10** itself. By setting this third bit in the register **126**, all processing may be stopped to force the screen to be background only.

The composer **124**, in processing its inputs, determines for any given video dot to be displayed on the monitor screen what its intensity will be, i.e., background, low intensity, or high intensity. The composer **124** is implemented by parallel decoding nand gates, as disclosed further herein, to represent the following functions. If a cursor is being displayed, then the intensity of the cursor overrides; high intensity cursor forces high intensity over a low intensity cursor; with no cursor being displayed, the video is displayed with whatever intensity is specified; and where video signals are not generated for display, the composer **124** displays the background.

The high intensity signals generated by the composer **124** are inputted to the shift register **116** where upon high intensity video signals are shifted out for display on the monitor screen. The low intensity signals generated by the composer **124** are inputted to the shift register **118** whereupon low intensity video signals are shifted out for display. Each of the registers receives two lines of video information, odd and even video respectively. The lines of video are modified on a clock divided by two basis. The clock/two input controls whether parallel loading or shifting occurs with respect to the registers **116** and **118.** The direct clock signal is an input to the shift registers **116** and **118** so that they may perform a function of alternately loading and shifting the odd and even video thus serializing the two video input signals into the final output. The shift registers **116** and **118** are the only elements in the generator **10** that must run at the bit clock speed.

The output signals from the composer **124** are thus processed by the shift registers **116** and **118** to provide the video high and low intensity signals which are fed to the video mixer **14** in the form of logic levels on two separate lines. In the mixer **14** these logic levels, e.g. 0 to 5 volts, are converted into TV video voltage levels, e.g. 0 to 1 volt, which are suitable as an input to the CRT monitor **1.**

The cursor control elements **112** and **114** and the composer **124** are further described in U.S. Pat. application Ser. No. 418,505 filed on Nov. 23, 1973 and assigned to the assignee of the present invention.

An additional output, external select, is generated from the character generator **10** as an input to the video mixer **14.** This external select signal is in this preferred embodiment a single bit which provides the selection of either external video of character generator video for display on the monitor **1.** The bit is derived from the mode register **94** which is ultimately derived from the contents of the display list program.

By placing control for selecting the video source within the display list program, overlays and screen partitioning can be achieved. For example, a picture can be displayed with the character generator **10** selected in places to display labels and/or titles, or arbitrary areas can be used for display of external video while the remaining area used for text from the character. This feature is implemented as alluded to earlier in the specification and as shown in FIG. **5** by placing a mode change instruction between the display characters in the display list which is utilized by the mode register **94** to control the video processing. The external signal from the mode register **94** is thus utilized when received by the mixer **14.** An analog switch within the mixer **14** is controlled by this signal to determine whether external video or character generator video is sent to the monitor **1.**

The video mixer **14** may be any conventional video mixer capable of performing these functions. The mixer device contemplated in this preferred embodiment, however, is that disclosed in U.S. Pat. application Ser. No. 418,506, filed on Nov. 23, 1973 and assigned to the assignee of the present invention.

Generation of high quality video information for display on high resolution T.V. systems requires digital processing which "pushes" the speed of integrated circuits currently available. While the required speed of 40 MHZ can be achieved with available components, they are significantly more expensive and take up more space. The invention overcomes this difficulty by processing the odd and even video bits separately and simultaneously, as shown in FIGS. **5, 7,** and **10.** The video output is derived from a 16-bit computer word where the individual bits are labeled 0, 1, 2, . . . 14, 15 and are presented to the output in sequence, bits 0, 1, **2, . . . 14, 15** at a 40 MHZ rate. Internally, however, one shift register presents bits 0, 2, 4, . . . 12, 14 while the other present bits 1, 3, 5, . . . 13, 15, both at a 20 MHZ rate. This allows any other control logic, such as the width counter **90** to operate at 20 MHZ. The only restriction in this approach is that the character widths must be even values only.

The video is produced by synchronously extracting words from the output buffer **50.** These words contain the character description, intensity and video mixing information. The output buffer **50,** though, is loaded asynchronously with words from the font memory **20,** which describes the characters to be displayed. The basis cycle time for the system as described herein is 220ns, which time is set by the speed of the memory devices used for the display list and font memories **34** and **20.** With the organization of elements as described, the maximum video output rate is 40 MHZ, or one dot every 25ns. To simplify the combination of elements following the buffer **50,** characters have a defined width consisting of an even number of dots.

It has been assumed in this description of the preferred embodiment that the binary coded data to be processed is stored in the memories and registers of this system. As implied earlier in the specification, though, the computer may initially write all of the stored information into the system by conventional interfacing with these elements. The function of the computer in either situation is to provide an interface between the display system described herein and processors which utilize the display system. Or course, each of the processors may choose to select a differing text on the display screen or even different fonts of characters, such as Roman, bold face, and italic, in differing sizes. Also, each processor may want to define its own character set and to operate as though it had a display screen of its own.

The controlling computer would have a library of fonts stored on a small disc. The representation of a sub-font may be specified either (1) by keyboard commands to the controlling computer which call out representations from the library in case the processor using the terminal is not equipped for fonts; (2) by similar commands from the processor in case the processor is equipped to handle fonts but has no representations of its own; or (3) by explicit specification of dot matrices from the processor.

Obviously, many modifications of the present invention are possible in light of the above teaching. It is therefore to be understood that, in the scope of the appended claims, the invention may be practiced other than as specifically described.

What is claimed is:

**1.** A device for generating video characters to be displayed on a linear scan video display device, comprising:

first memory means for storing binary information representing a plurality of characters,

said first memory means being comprised of storage cells, each of which stores constant size font definition matrices;

means for processing the binary information stored in said first memory means to characters for display;

second memory means for storing instructions which control the generation of the binary information to be processed; and means coupled between said first and second memory means for indexing said first memory means such that variant sized matrices are provided to define the displayed characters on the display screen of the display device.

**2.** The device in claim **1** wherein is further included character dimensioning means responsive to instructions in said second memory means for providing variable widths of said characters and the variable spacing thereof.

**3.** The device of claim **1** wherein is further included means for segmenting the display raster such that segmented fields of characters are provided.

**4.** The device in claim **1** wherein is further included a third memory means connected in parallel with said first memory means for providing overlay characters for display.

**5.** The device in claim **1** wherein is further included a register means for storing character control information from said second memory means and wherein said first memory means is responsive to the output of said register means.

**6.** The device of claim **5** wherein is further included character dimensioning means responsive to the output of said register means for providing variable widths of said characters and the variable spacing thereof.

**7.** The device of claim **6** wherein said register means is responsive to the output of said dimensioning means at the beginning of the next memory cycle associated with said memory means.

**8.** The device of claim **7** wherein is further included means for segmenting the display raster such that segmented fields of characters are provided.

**9.** The device of claim **8** wherein is further included a third memory means connected in parallel with said first memory means for providing overlay characters for display.

**10.** A device for generating video characters to be displayed on a linear scan video display device, comprising:

first memory means for storing binary information representing a plurality of characters,

said first memory means being comprised of storage cells, each of which stores constant size font definition matrices;

means for processing the binary information stored in said memory means to characters for display;

second memory means for storing instructions which control the generation of the binary information to be processed; and

register means coupled to said first and second memory means for storing character control information from said second memory means,

said first storage means being responsive to the output of said register means, such that variant sized matrices are provided to define the displayed characters on the display screen of the display device.

**11.** The device in claim **10** wherein is further included character dimensioning means responsive to the output of said register means for providing variable widths of said characters and the variable spacing thereof.

**12.** The device in claim **11** wherein said register means is responsive to the output of said dimensioning means at the beginning of the next memory cycle associated with said memory means.

**13.** The device of claim **12** wherein is further included means for segmenting the display raster such that segmented fields of characters are provided.

**14.** The device of claim **13** wherein is further included a third memory means connected in parallel with said first memory means for providing overlay characters for display.

**15.** The device in claim **1** wherein said indexing means includes means for expanding font definition matrices vertically to provide an effective height for a given character greater than the cell height for that character.

**16.** The device of claim **15** wherein said indexing means further includes means for extending the font definition matrices horizontally to provide an effective width for a given character greater than the cell width for that character.

**17.** The device in claim **16** wherein said extension means is responsive to the output of said first memory means as well as said second memory means for the generation of extension information to said first memory means whereby indexing is achieved.

**18.** The device of claim **10** wherein said register means includes means for expanding font definition matrices vertically to provide an effective height for a given character greater than the cell height for that character.

**19.** The device of claim **18** wherein said register means includes means for extending the font definition matrices horizontally to provide an effective width for a given character greater than the cell width for that character.

**20.** The device of claim **19** wherein said extension means is responsive to the output of said first memory means as well as said second memory means for the generation of extension information to said first memory means whereby variant sized matrices are provided for display.

**21.** A device for generating video characters to be displayed on a linear scan video display device, comprising:

first memory means for storing binary information representing a plurality of characters,

said first memory means being comprised of storage cells, each of which stores constant size font definition matrices;

means for processing the binary information stored in said memory means to characters for display;

second memory means for storing instructions which control the generation of the binary information to be processed;

register means coupled to said first and second memory means for storing character control information from said second memory means,

said register means including means for extending the font definition matrices of said first memory means horizontally to provide an effective width for a given character greater than the cell width for that character,

said extension means being responsive to the character control information and the output of said first memory means for the generation of extension information to said first memory means such that variable character widths and variable spacing for characters may be provided.

**22.** The device of claim **21** wherein said register means further includes means for expanding the font definition matrices vertically to provide an effective height for a given character greater than the cell height for that character.

**23.** The device of claim **22** wherein is further included a third memory means connected in parallel with said first memory means for providing overlay characters for display, said third memory means being

responsive to said extension means for providing character extension overlays.

**24.** The device of claim **22** wherein is further included means for segmenting the display raster such that segmented fields of characters are provided.

* * * * *