

[54] CATHODE RAY TUBE PRESENTATION OF CHARACTERS IN MATRIX FORM FROM STORED DATA AUGMENTED BY INTERPOLATION

[75] Inventor: Paul Allen King, Jr., Cambridge, Mass.

[73] Assignee: Massachusetts Institute of Technology, Cambridge, Mass.

[22] Filed: Aug. 17, 1972

[21] Appl. No.: 281,596

Related U.S. Application Data

[63] Continuation of Ser. No. 50,641, June 29, 1970.

[52] U.S. Cl. 340/324 AD, 178/DIG. 3

[51] Int. Cl. G06f 3/14

[58] Field of Search 340/324 AD; 178/DIG. 3

[56] References Cited

UNITED STATES PATENTS

3,396,377 8/1968 Strout 340/324 AD
3,422,737 1/1969 Bailey, Jr. 340/324 AD

3,573,789 4/1971 Sharp et al. 178/DIG. 3

Primary Examiner—Thomas B. Habecker

Assistant Examiner—Marshall M. Curtis

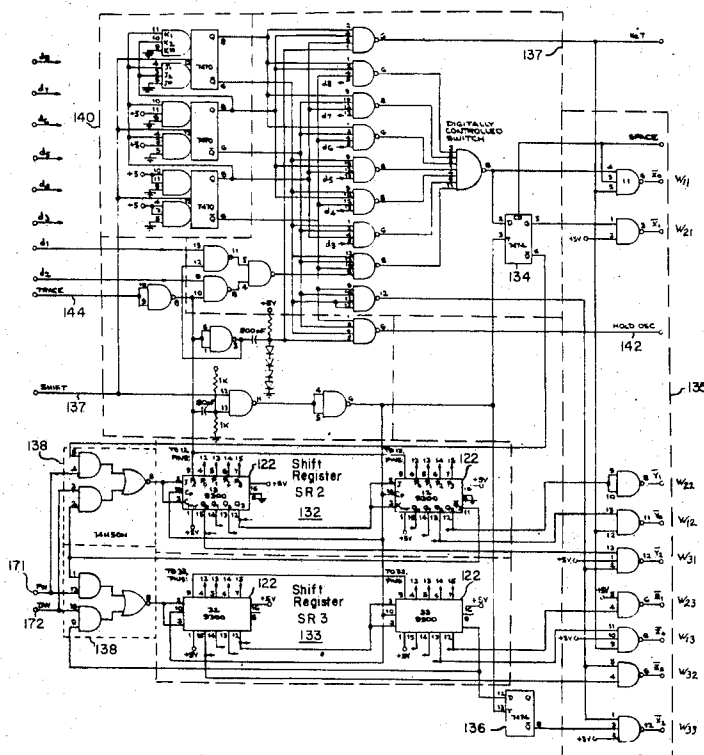
Attorney—Thomas Cooch et al. and Arthur A. Smith, Jr.

[57]

ABSTRACT

Apparatus is disclosed for displaying a selected character on the face of a cathode ray tube at a particular location from information previously stored in a memory. The cathode ray tube is swept in a raster scan with a superimposed sinusoidal deflection in the y-axis direction. The intensity modulation which produces a dot-like representation of a character occurs at a particular time during the scan depending upon the desired location of the character. The intensity modulation as a function of time is unique for each character and occurs along the sinusoidal deflection. The required storage capacity of the memory storing the character information is reduced by an interpolation technique.

7 Claims, 34 Drawing Figures



PATENTED JAN 15 1974

3,786,478

SHEET 01 OF 26

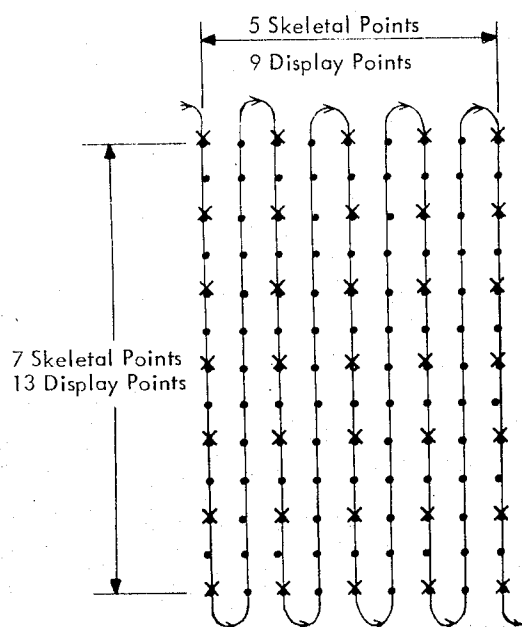


FIG. 1

INVENTOR
PAUL ALLEN KING, JR.

BY

Thos. M. Hester

ATTORNEY

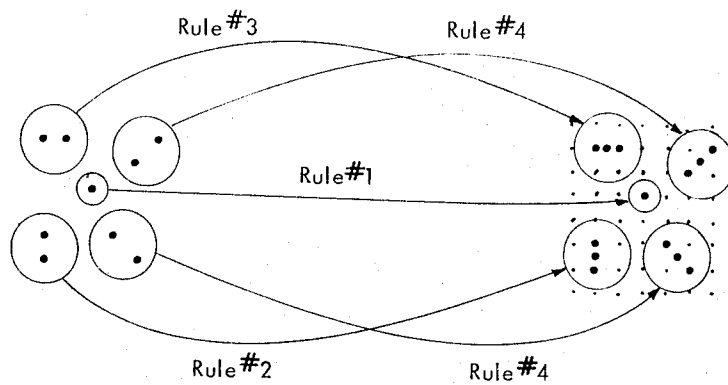


FIG. 2

INVENTOR:

PAUL ALLEN KING, JR.

BY

Martin M. Hinton
ATTORNEY

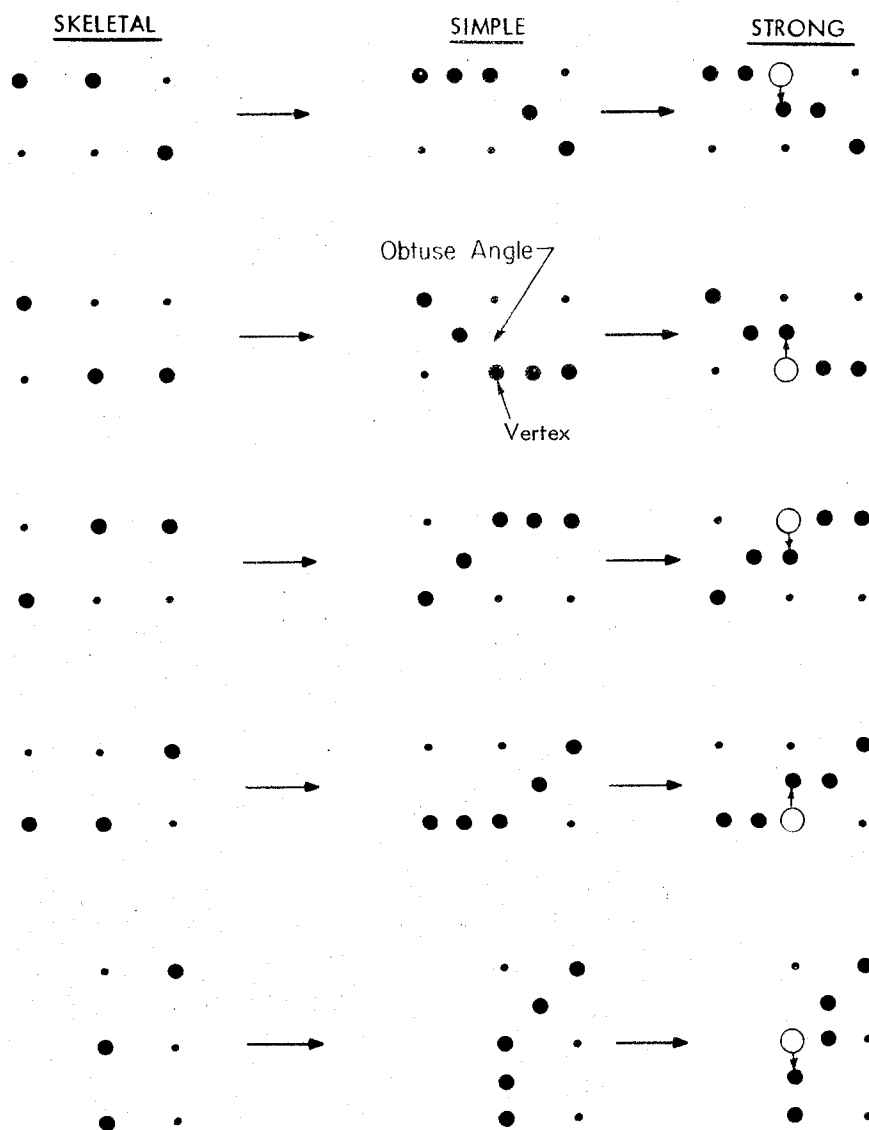


FIG. 3

INVENTOR:

PAUL ALLEN KING, JR.

BY *Trustin M. Fanta*
ATTORNEY

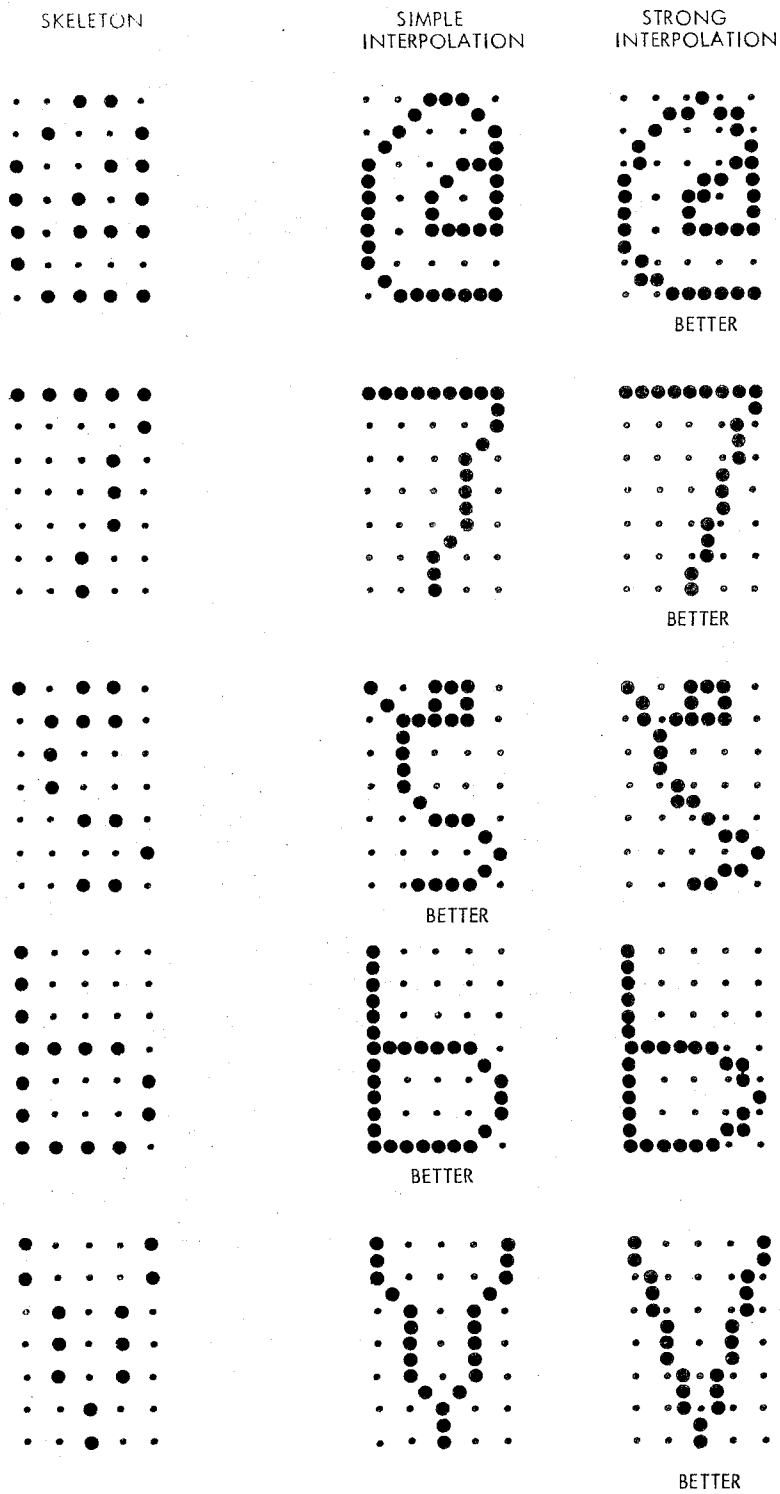


FIG. 4

INVENTOR:
 PAUL ALLEN KING, JR.
 BY *Walter D. Hunter*
 ATTORNEY

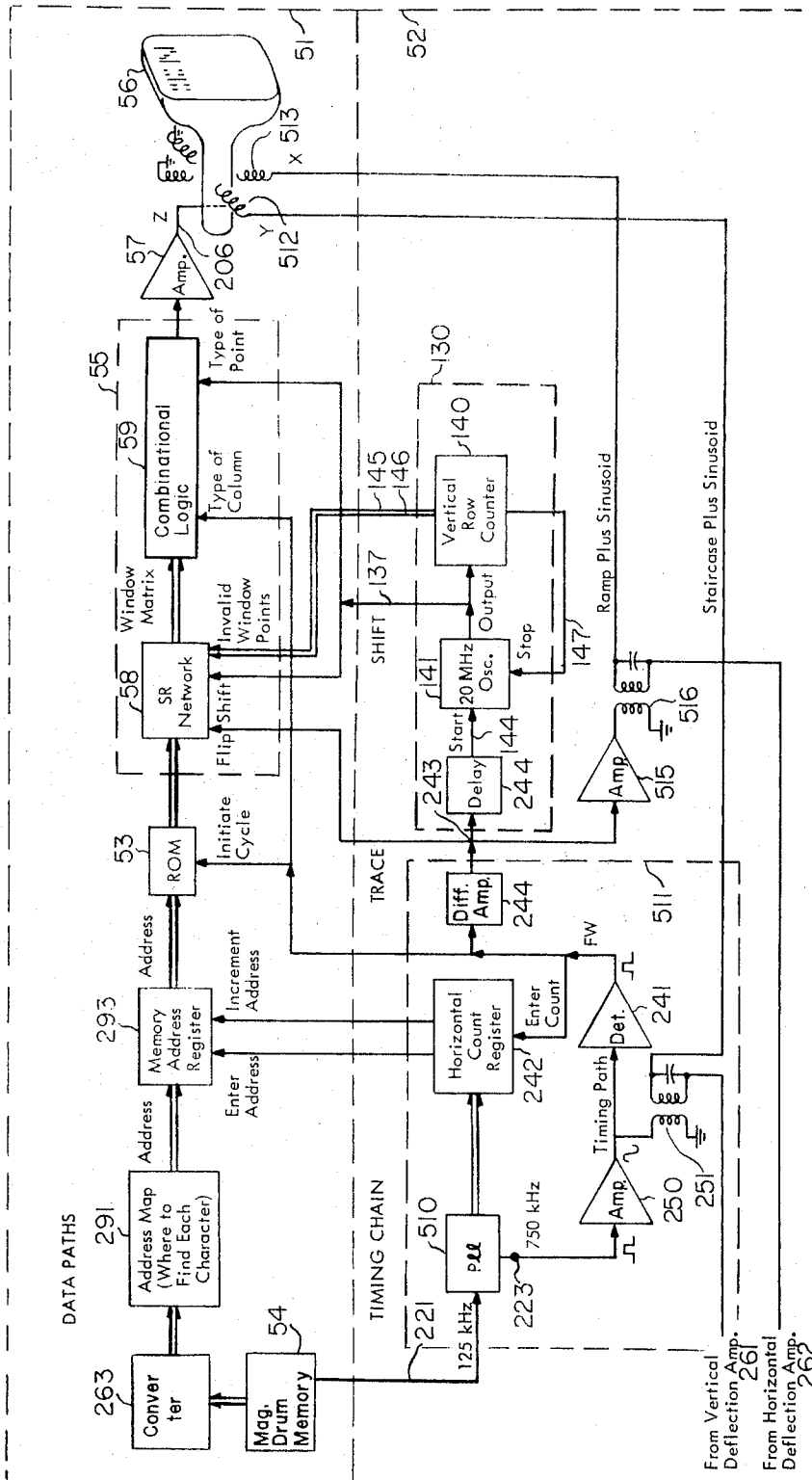


FIG. 5

INVENTOR:

PAUL ALLEN KING, JR.

Dr. *Hyacinthoides*

ATTORNEY

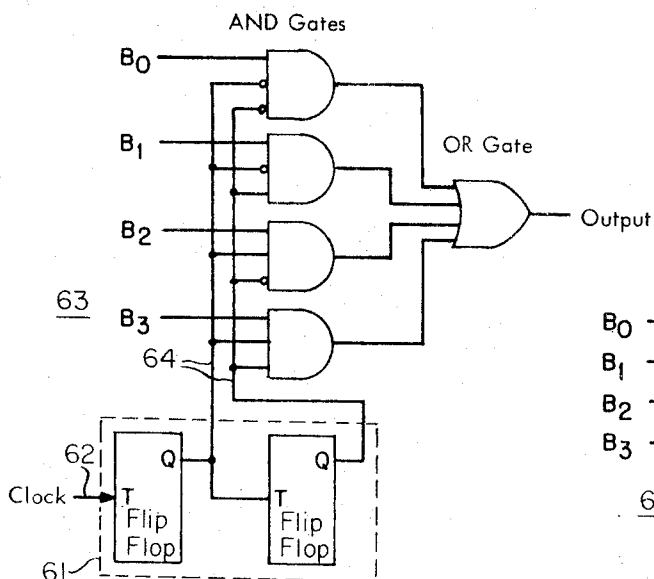


FIG. 6A

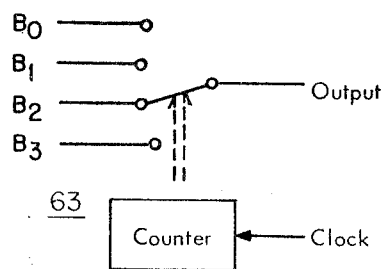


FIG. 6B

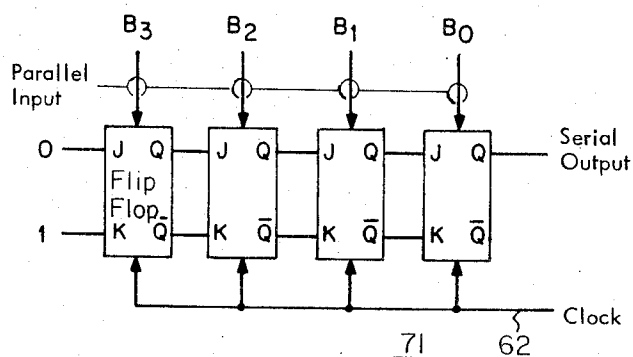


FIG. 7

INVENTOR:
PAUL ALLEN KING, JR.
BY *Martin J. Fawcett*
ATTORNEY

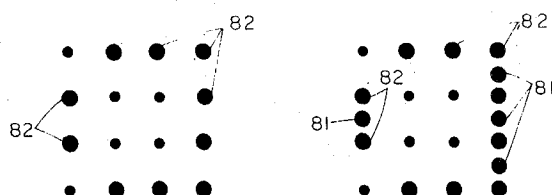


FIG. 8

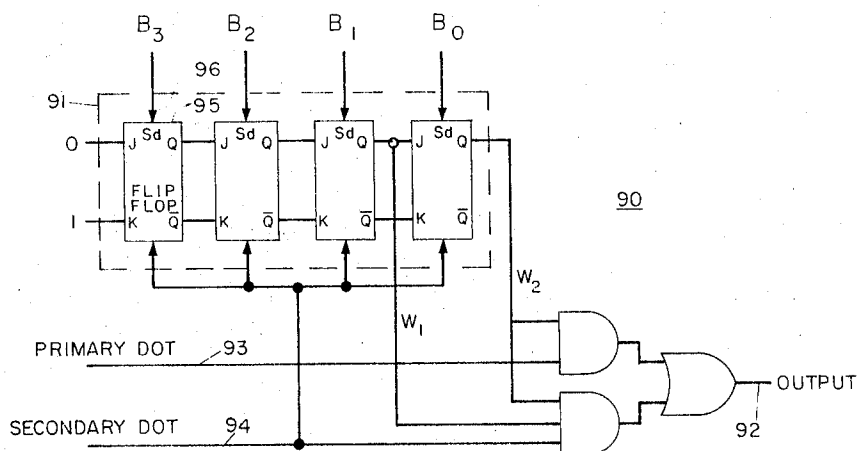


FIG. 9

INVENTOR:
PAUL ALLEN KING, JR.

BY *Paul Allen King, Jr.*
ATTORNEY

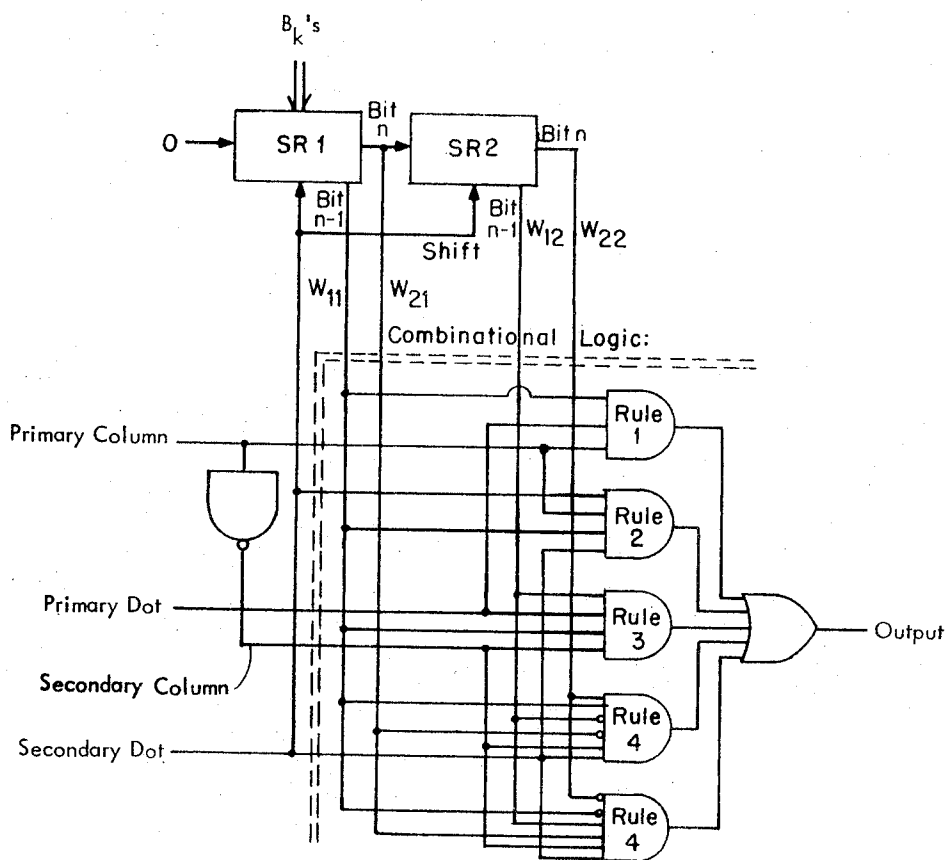


FIG. 10

INVENTOR:

PAUL ALLEN KING, JR.

BY *[Signature]*

ATTORNEY

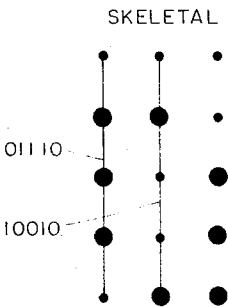


FIG. IIA

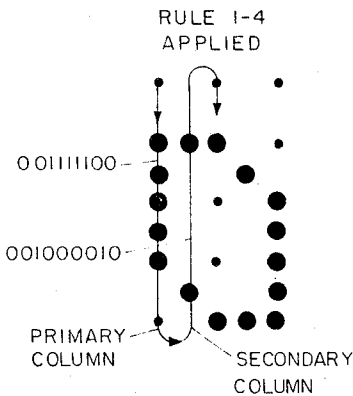


FIG. IIB

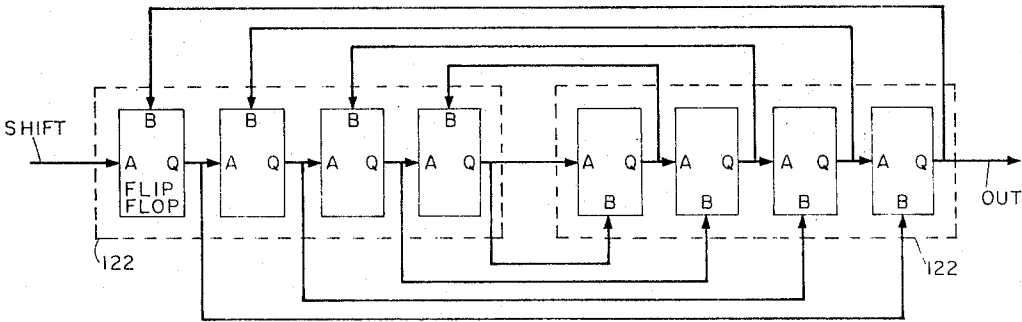


FIG. I2

INVENTOR:
PAUL ALLEN KING, JR.
BY *[Signature]*
ATTORNEY

Type column	Contents of SR1	Contents of SR2	Window matrix	Output time	Output value	Rule used
example:	ABCDE	FGHIJ	JE ID			
	load word					
Primary:						
	01110	φφφφφ	φ0 φ1	1.0	0	-
	φ0111	0φφφφ	φ1 φ1	1.5	0	-
	φφ011	10φφφ	φ1 φ1	2.0	1	#1
	φφφ01	110φφ	φ1 φ1	2.5	1	#2
	φφφφ0	1110φ	φ1 φ0	3.0	1	#1
	φφφφφ	01110	φ0 φ0	3.5	1	#2
				4.0	1	#1
				4.5	0	-
				5.0	0	-
	load new word					
Secondary:						
	10010	01110	00 11	6.0	0	-
	φ1001	00111	11 10	6.5	0	-
	φφ100	10011	10 10	7.0	1	#3
	φφφ10	01001	10 01	7.5	0	-
	φφφφ1	00100	01 0φ	8.0	0	-
	φφφφφ	01001		8.5	0	-
				9.0	0	-
				9.5	1	#4
				10.0	0	-

FIG. II c

INVENTOR:

PAUL ALLEN KING, JR.

BY *Martin J. Santa*

ATTORNEY

PRIMARY COLUMN, PRIMARY DOT

$W_{11} \bullet \bullet W_{12}$
 $W_{21} \bullet \bullet W_{22}$

RULE # 1

PRIMARY COLUMN, SECONDARY DOT

$\bullet \bullet$
 $\bullet \bullet$

RULE # 2

SECONDARY COLUMN, PRIMARY DOT

$\bullet \bullet$
 $\bullet \bullet$

RULE # 3

SECONDARY COLUMN, SECONDARY DOT

$\bullet \circ \circ \bullet$
 $\circ \bullet \bullet \circ$

RULE # 4

FIG. IID

$\bullet W_{12} \bullet W_{13}$

$\bullet W_{22} \bullet W_{23}$

$\bullet W_{31} \bullet W_{32}$

SECONDARY SCAN

FIG. 16 (A)

$\bullet W_{11} \bullet W_{12} \bullet W_{13}$

$\bullet W_{21} \bullet W_{22} \bullet W_{23}$

$\bullet W_{31} \bullet W_{32} \bullet W_{33}$

PRIMARY SCAN

FIG. 16 (B)

INVENTOR:

PAUL ALLEN KING, JR.

BY *Thorton T. Harte*
ATTORNEY

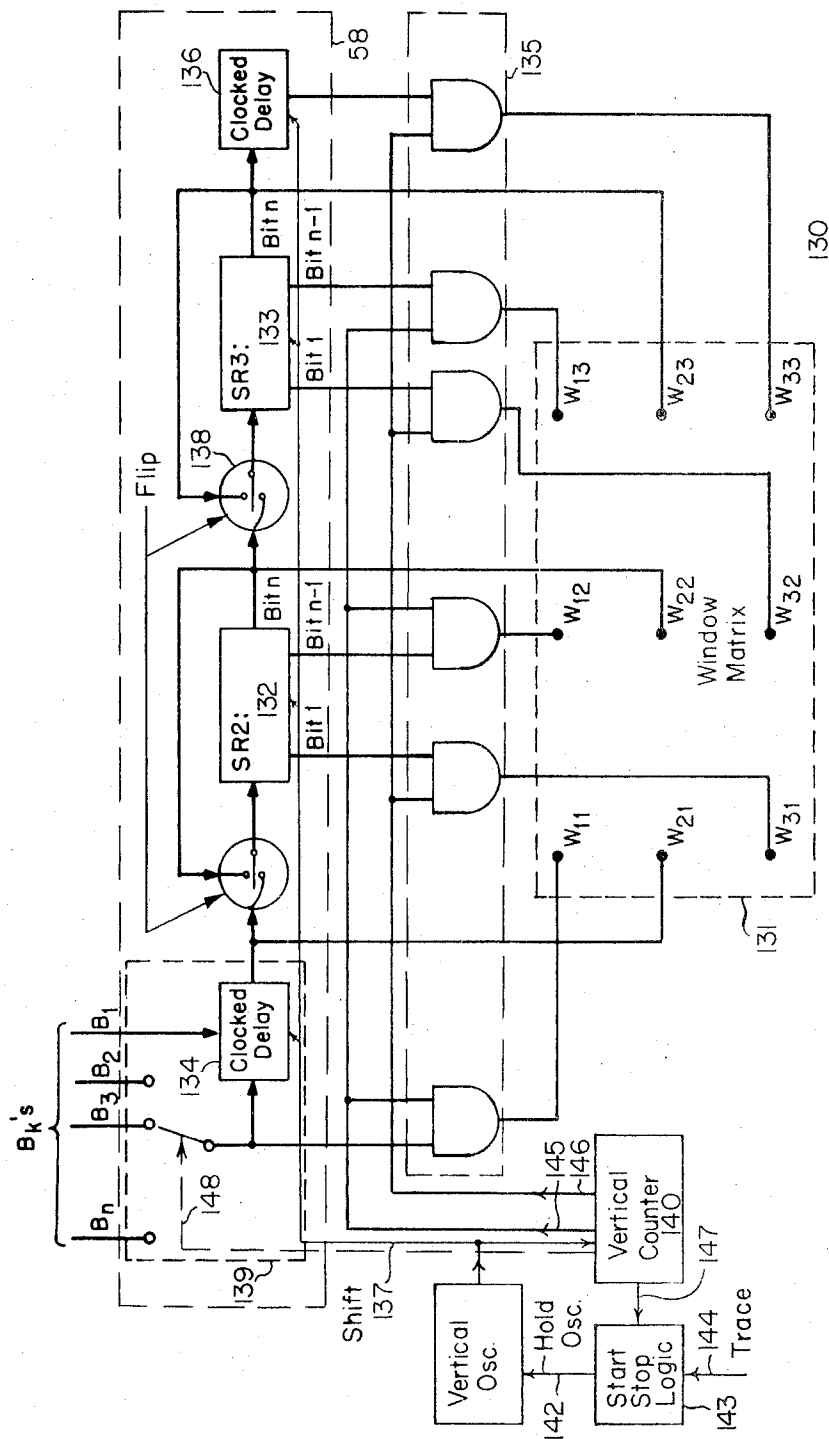


FIG. 13

INVENTOR:

PAUL ALLEN KING, JR.

BY *[Signature]*

ATTORNEY

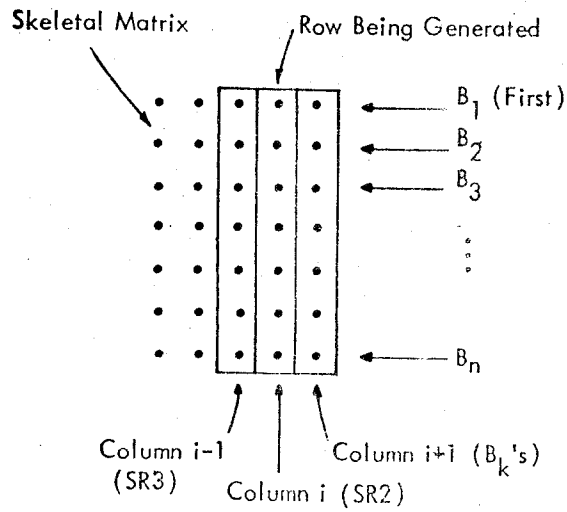


FIG. 14

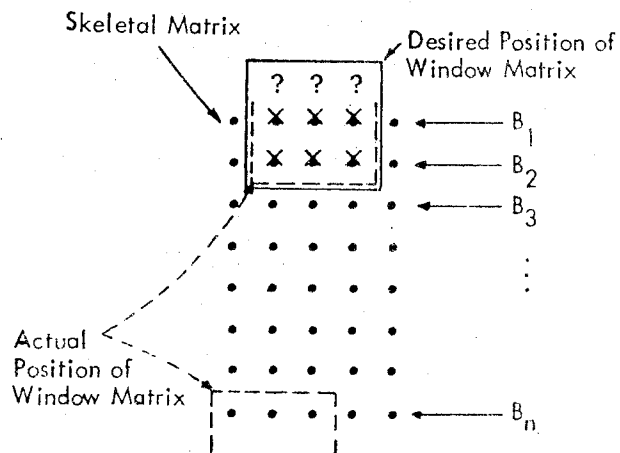


FIG. 15

INVENTOR

PAUL ALLEN KING, JR.

BY *Paul Allen King, Jr.*

ATTORNEY

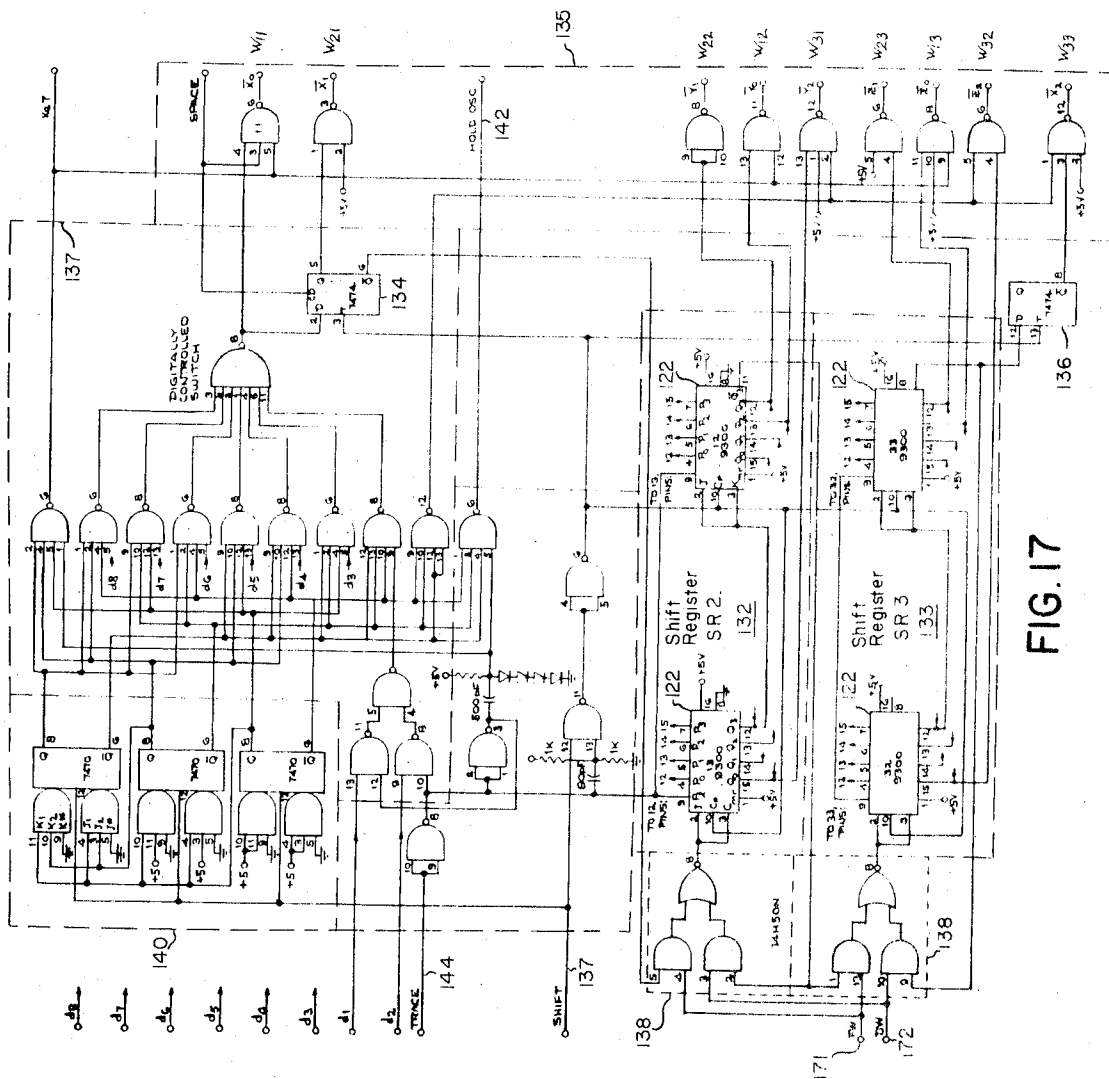


FIG. 17

INVENTOR:

PAUL ALLEN KING, JR.

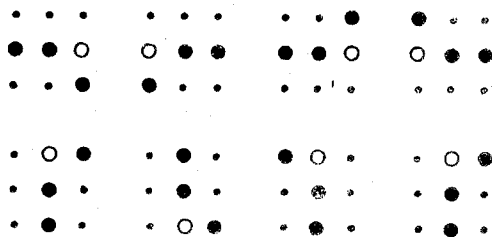
BY

ATTORNEY

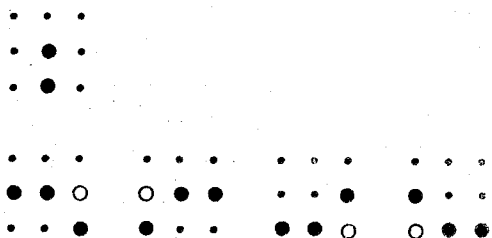
Primary Column, Primary Dot is on only if the Pattern



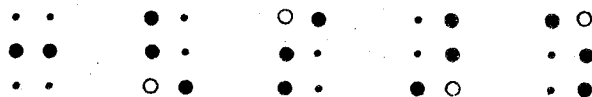
is Present in the Three by Three, and None of the Following Patterns are Present



Primary Column, Secondary Dot



Secondary Column, Primary Dot



Secondary Column, Secondary Dot

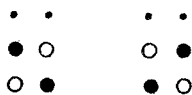


FIG. 18

INVENTOR:

PAUL ALLEN KING, JR.

BY *[Signature]*

ATTORNEY

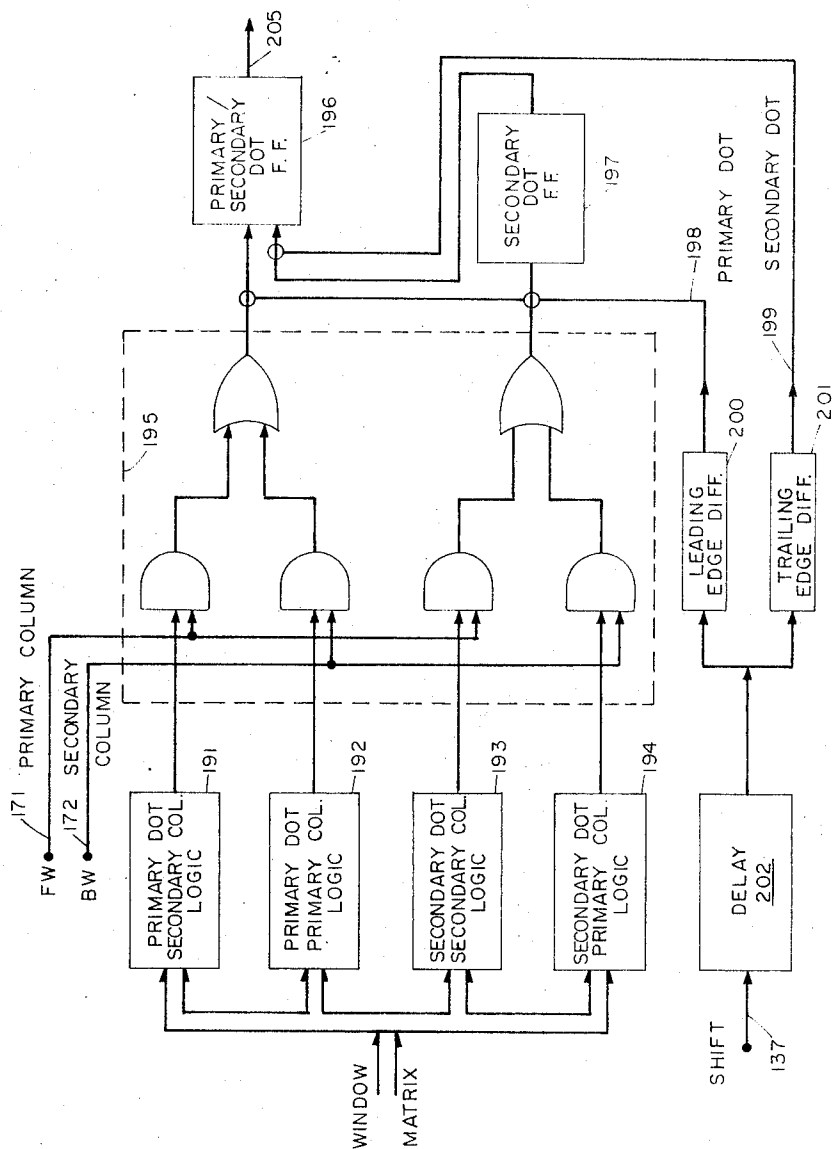


FIG. 19

INVENTOR:

PAUL ALLEN KING, JR.

BY

ATTORNEY

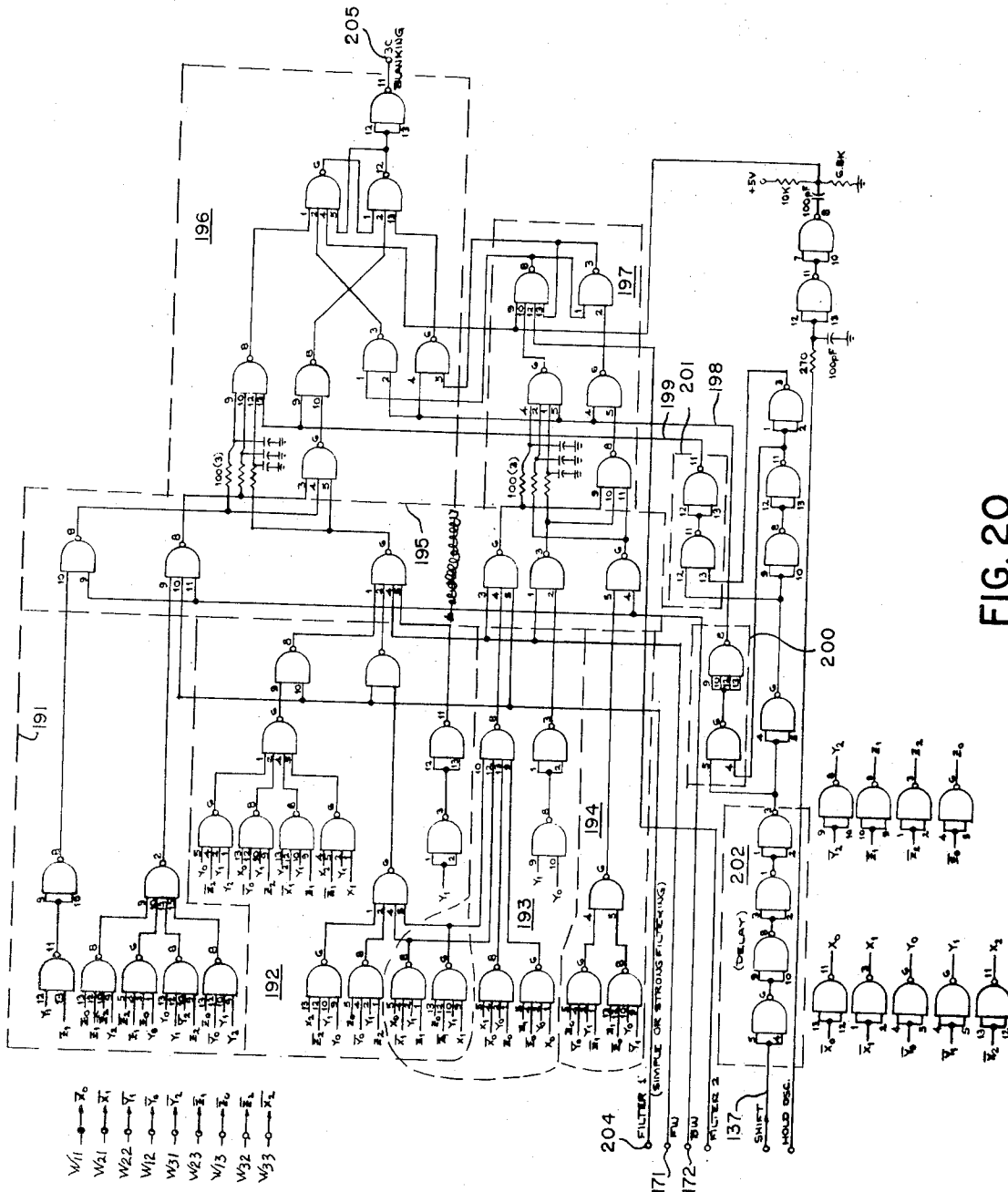
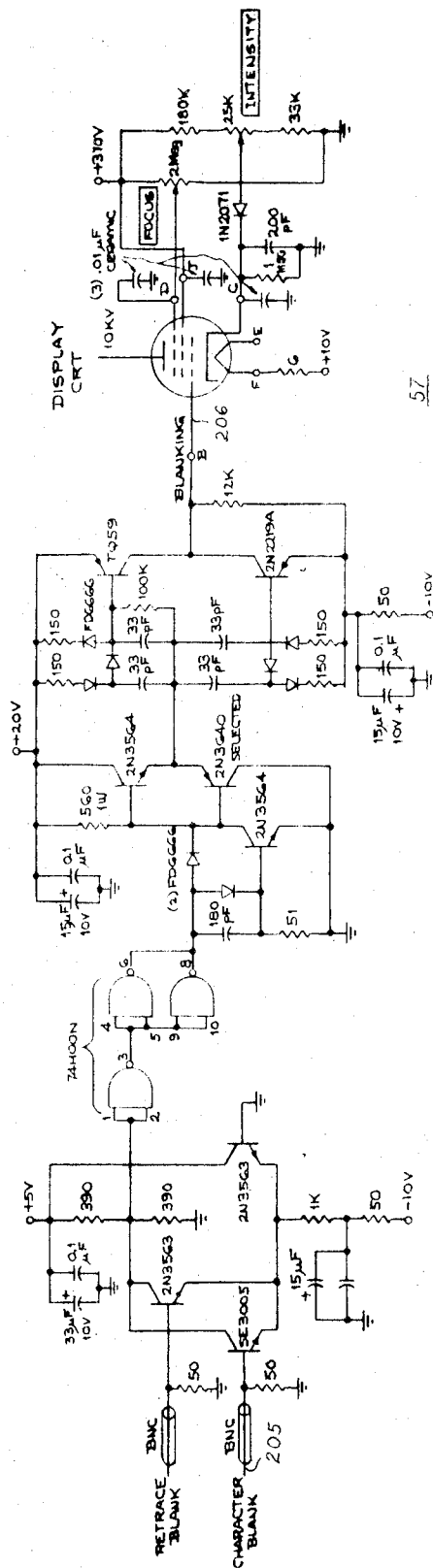


FIG. 20

INVENTOR:
 PAUL ALLEN KING, JR.
 BY *Martin M. Foster*
 ATTORNEY



57

FIG. 21

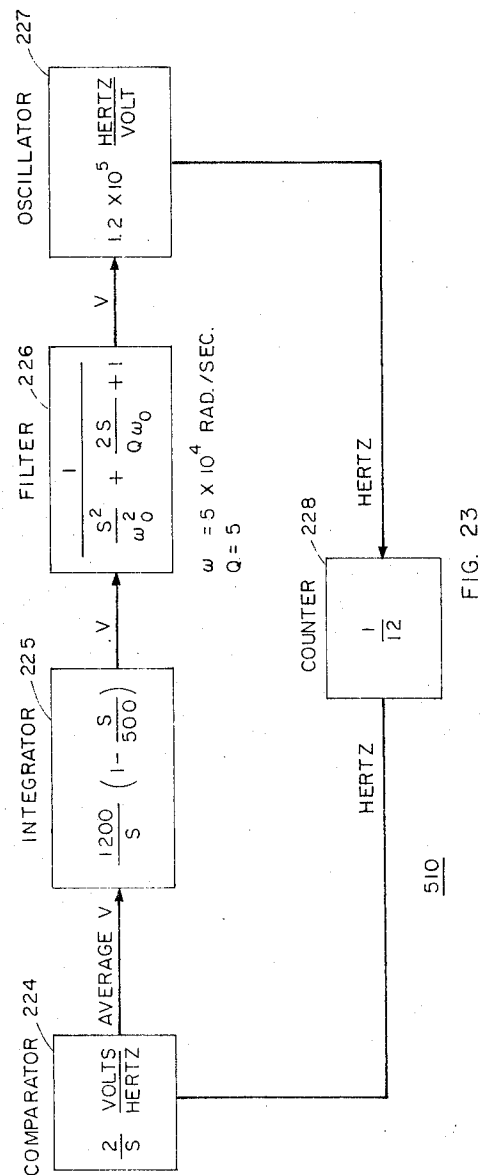
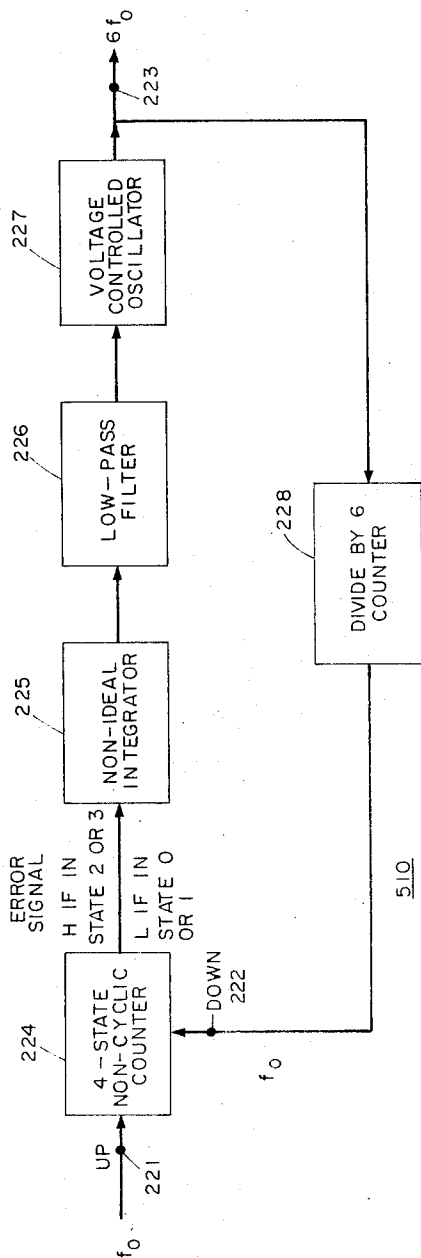
NOTES:

- 1. YOKE - CELCO
F4920-5660
- 2. SYLVANIA CRT
GUN - 6NPA
BOTTLE - MAUP4
PHOSPHOR - P31

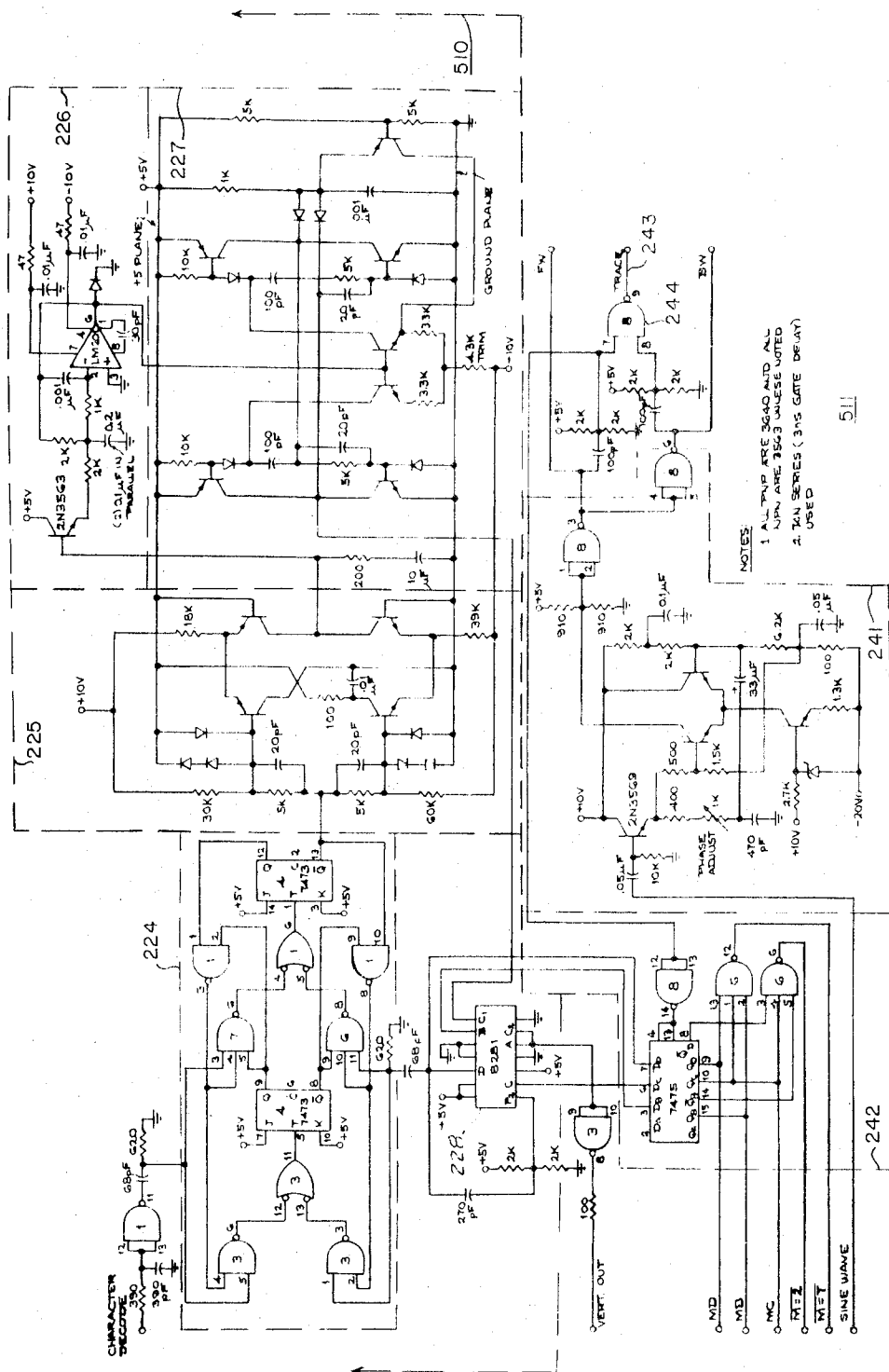
- K 0 → +70V
- R 0 → -10V
- S 0 → +5V
- T 0 → +10
- U 0 → +20

INVENTOR:
PAUL ALLEN KING, JR.

BY *[Signature]*
ATTORNEY



INVENTOR:
 PAUL ALLEN KING, JR.
 BY *[Signature]*
 ATTORNEY



INVENTOR:
PAUL ALLEN KING, JR.
BY
ATTORNEY

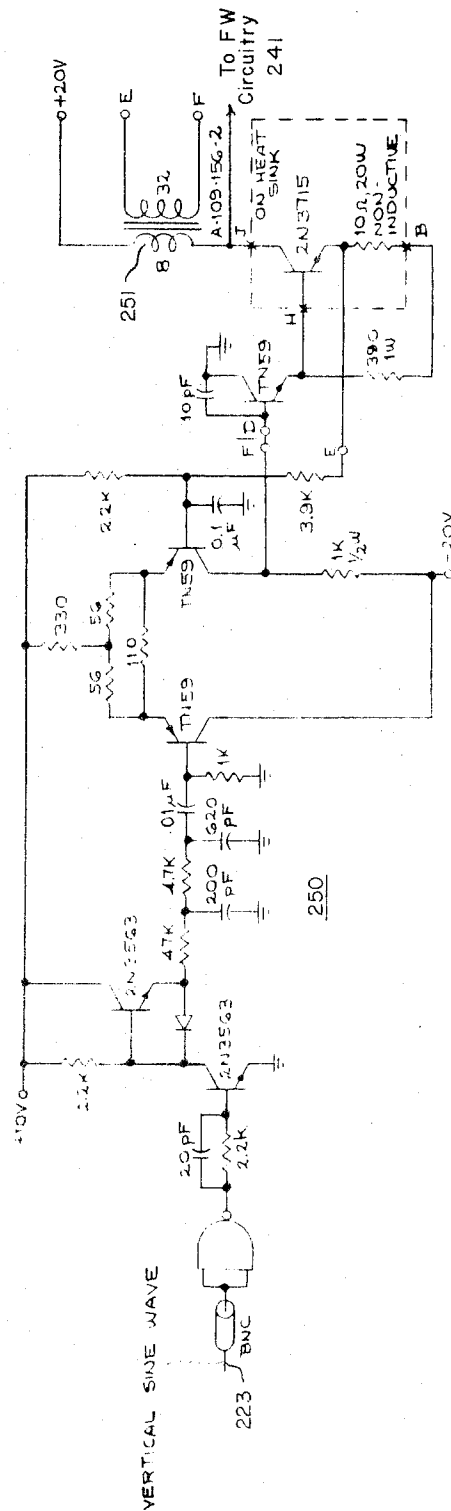


FIG. 25

INVENTOR:

PAUL ALLEN KING, JR.

BY *James H. Smith*
ATTORNEY

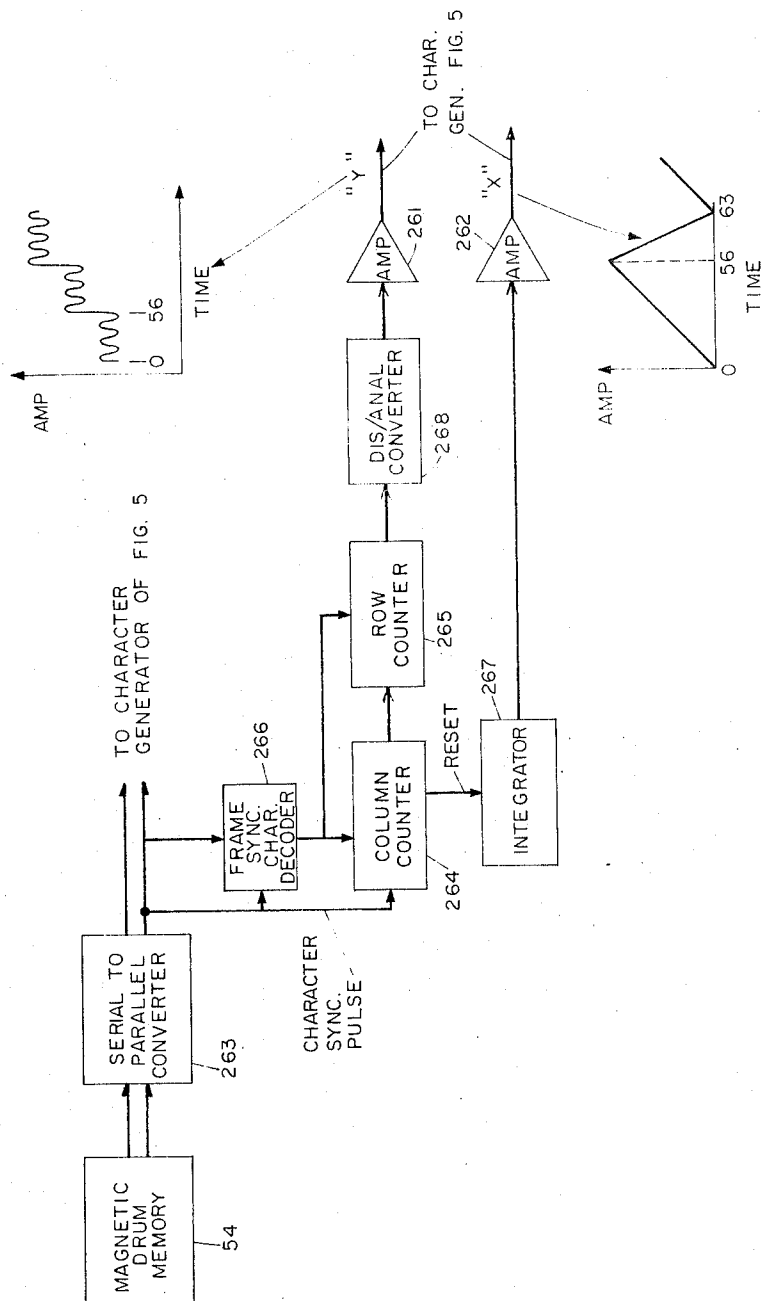


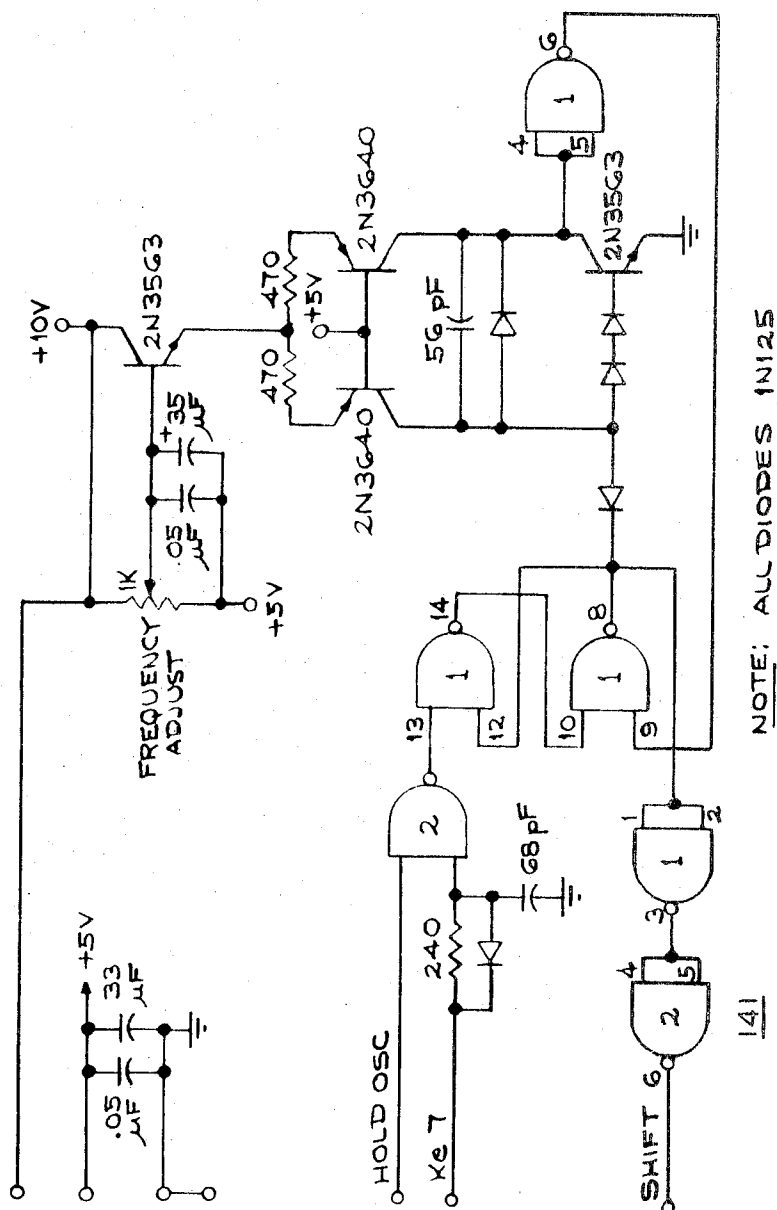
FIG. 26

INVENTOR:

PAUL ALLEN KING, JR.

BY

ATTORNEY



INVENTOR:
PAUL ALLEN KING, JR.

ASCII Code	Rope Address	Data	Character	Code
100	0000	0011110	@	6764321
	0001	0100000		
	0002	1001101		
	0003	1010101		
	0004	0111101	Space	01000000
40	0005	0000000		
	0006	0000000		
	0007	0000000		
	0008	0000000	/	00100000
	0009	0000000		
140	0010	0000000		
	0011	1000000		
	0012	0100000	Key word address	01100000
	0013	0010000		
	0014	0000000		
	0015	Key word address		
101	0016	0001111	A	01000001
	0017	0010100		
	0018	0001100		
	0019	0010100		
	0020	0001111	!	00100001
141	0021	0000000		
	0022	0000000		
	0023	1111101		
	0024	0000000	a	01100001
	0025	0000000		
241	0026	0000010		
	0027	0010101		
	0028	0010101	Key word address	
	0029	0010101		
	0030	0001111		
	0031	Key word address		
	0032	etc		
	0033			

First
16 word
block

Second
16 word
block

FIG. 28

INVENTOR:

PAUL ALLEN KING, JR.

BY

ATTORNEY

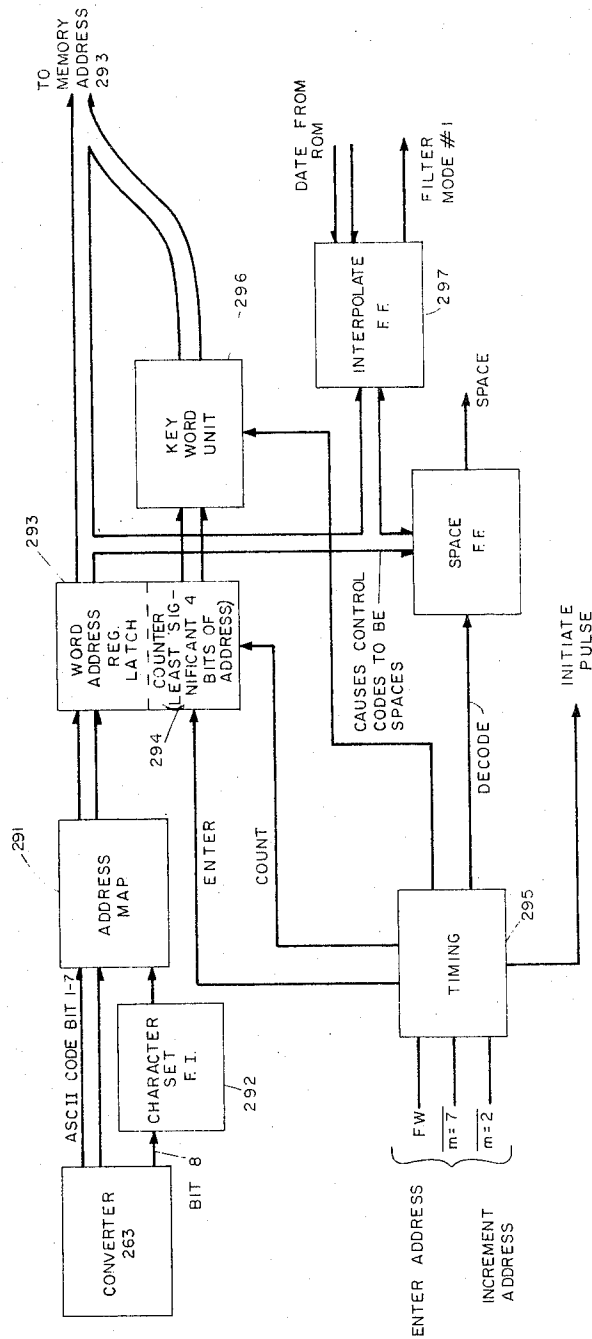


FIG. 29

INVENTOR:

PAUL ALLEN KING, JR.

BY

ATTORNEY

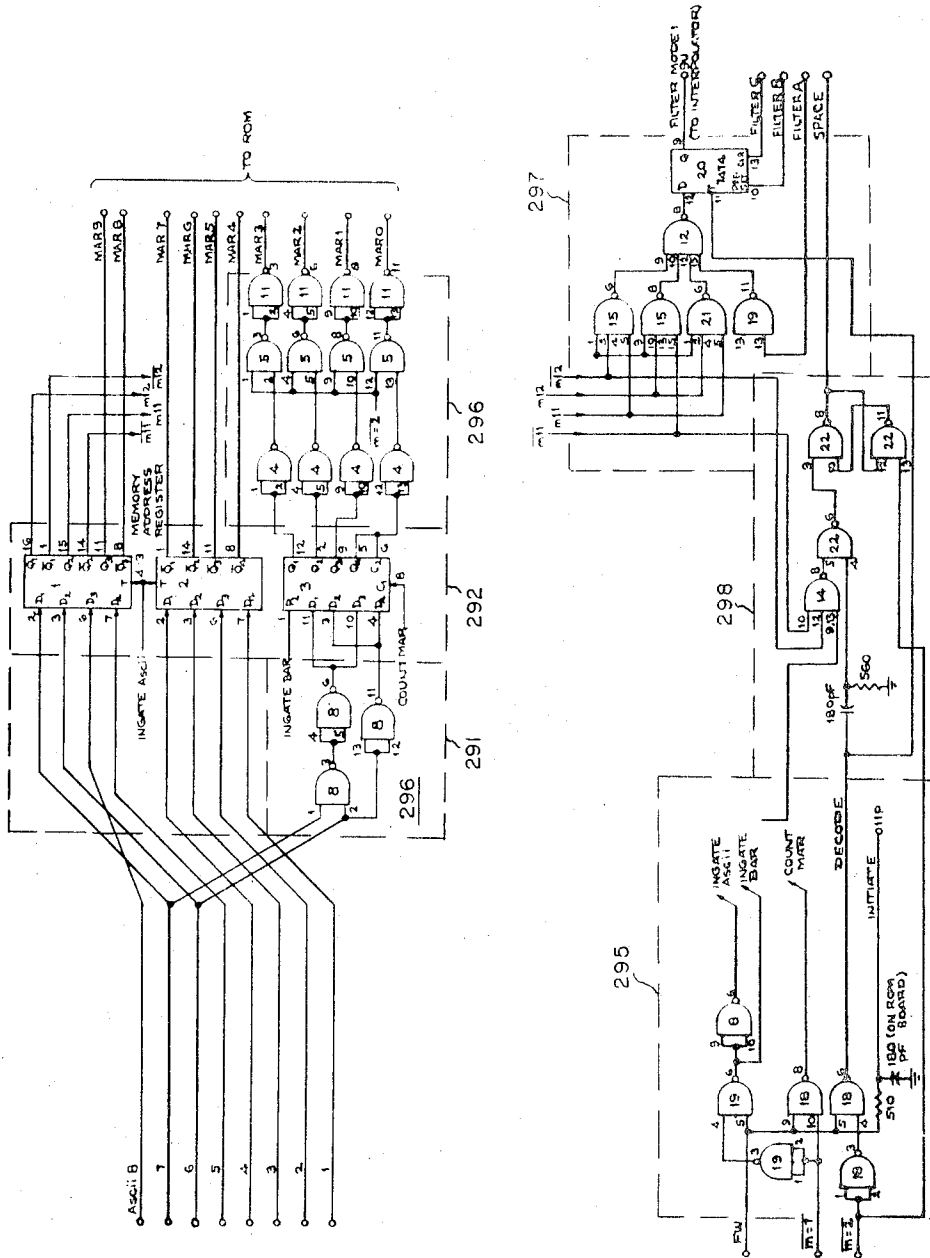


FIG. 30

INVENTOR:

PAUL ALLEN KING, JR.

BY

ATTORNEY

CATHODE RAY TUBE PRESENTATION OF CHARACTERS IN MATRIX FORM FROM STORED DATA AUGMENTED BY INTERPOLATION

This is a continuation, of application Ser. No. 50,641, filed June 29, 1970.

This invention relates to a character generator and in particular to a generator which produces characters by intensity modulation of a small sinusoidally deflected cathode ray beam in a raster scan. The intensity modulation is a time sequence of z axis intensifying pulses unique for each character to be presented.

The character generator of this invention displays characters in the form of intensity modulated spots on a large screen cathode-ray tube (CRT). The generator has the capacity of displaying approximately 1,700 characters — 31 lines of 56 characters per line. The character set includes both upper and lower case English letters (the standard 96 character ASCII code). The set also includes an additional 96 symbols including Greek letters and mathematical symbols. The display system has the provision for selective erasure, to facilitate editing and making corrections. Since the information with respect to each character's form and location on the face of the CRT is determined by its code located at a particular point on a rotating magnetic drum memory the character's form and location may be inserted or changed in the memory by the technique widely known by the skilled in the art of using a keyboard to provide an ASCII coded character at a particular point on the rotating magnetic drum with the aid of a computer and a cursor. Since the invention is not directed to apparatus for changing or inserting character information onto the rotating drum memory which is old in the art, it will be assumed that the information is stored in the drum memory by a suitable means and that viewing on a CRT that which is stored is the problem to which the invention is applied. As an example of apparatus for providing and changing stored information in a drum memory suitable for use in conjunction with this invention, see the report of D. R. Haring, "The Augmented-Catalog Console for Project INTREX (Part II)," ESL-TM-410, Electronic Systems Laboratory, Mass. Inst. of Tech.

All information displayed on the CRT is stored on the magnetic drum. The data on the drum passes to the character generator and is displayed on a screen in the following manner. A format generator produces a raster of over 1,700 character positions every 17 milliseconds. The generator is synchronized to the magnetic drum so that one complete revolution of the drum corresponds to one complete raster on the CRT. Consequently, drum addresses correspond to specific positions on the CRT screen, resulting in a savings of hardware. Note that this scheme requires the character generator illuminate a whole character on the CRT screen every time the drum sends one (every 8.5 μ s). The character generator is constantly a slave to the codes coming from the drum.

Much of the expense of a large capacity refreshed alpha-numeric display is taken up by the CRT and its associated deflection circuits. To reduce this cost, an inexpensive television picture tube is being used. The tube is magnetically deflected, with a deflection-system settling time of about 10 μ s. This means that the beam cannot be moved quickly to an arbitrary point on the screen since time has to be allowed for the beam to settle.

The solution is to selectively intensify a fixed scan pattern. The pattern, which was chosen because it is immune to the settling time of the beam, is generated by adding a sine wave to the vertical deflection signal, and applying a ramp to the horizontal deflection. Both patterns can be implemented easily with hardware because they both have very narrow bandwidth.

The vertical scan pattern is immune to settling time because in the sinusoidal-steady-state analysis, a time delay is merely a phase lag. This lag, if it is constant, can be corrected in theory by an appropriate phase advance of the signal to the deflection circuit, although in practice, it is effected by a phase delay in an alternate path. The ramp, which is applied to the horizontal deflection, sweeps a full line of 56 characters without stopping. Since ample time (about 20 μ s) is allowed at the beginning of each line for the beam to stabilize its transient response, limited horizontal deflection bandwidth is tolerable.

The sinusoidal scan pattern has the advantages of simplicity and ease of implementation. It is probably the only effective way in which one can display characters with deflection circuits that require 10 μ s to settle. Thus vector generators (which have variable scan patterns) and the type of dot matrix generators which rely on digital spot positioning are truly not feasible.

Constrained to using the sinusoidal scan pattern, there are two major ways to generate the necessary blanking signal. One way involves the prior art techniques of electrically scanning an image of the character set which has been physically constructed in the device. The other method, as in this invention, stores the intensity patterns in a digital read-only memory. The patterns for both are, of course, displayed on the sinusoidal scan.

There are three types of prior art scanning generators, the flying spot scanner, the monoscope, and the vidicon tube. The weaknesses of these scanning techniques are well known to those skilled in the art and in large measure do not exist in the digitally stored image of this invention.

SUMMARY OF THE INVENTION

A magnetic memory drum is selected which has sufficient capacity to store sequentially all the character codes for each character and its position on the face of a cathode ray tube. The cathode ray tube is linearly swept across its face in the x direction for each incremental deflection in the y direction to produce a raster. One rotation of the drum of the magnetic memory corresponds to a period of the vertical scan frequency or one complete raster. The vertical sweep and the horizontal sweep of the cathode ray tube are synchronized to the rotation of the drum magnetic memory so that a particular location on the circumference of the drum corresponds to a particular location of the beam on the face of the tube.

At each location on the circumference of the drum where a character may be stored there is a pulse which is utilized to synchronize a phase-locked oscillator. The oscillator provides a high frequency sine wave which is applied to the vertical deflection circuitry to provide a sinusoidal scan superimposed on the stepped y-scan. If a character is desired to be written on the face of the cathode ray tube at a particular location a code representing that character is written into the drum magnetic memory at a location corresponding to the desired

point on the cathode ray tube. Thus, when in its rotation the drum reaches that particular location the code corresponding to that character is read out to decoding circuitry which selects that desired character from the many characters stored in a conventional rope core memory, a read only memory.

Each drum rotation provides a pulse for each frame and a pulse at each character location along its circumference. The pulses from the character locations are fed into a phase locked loop oscillator which produces a frequency which is 6 times the pulse repetition frequency so that there are twelve sweeps of the beam per character space, 9 of which are used in producing the character and 3 of which are the character space. The phase locked loop oscillator produces a square wave output. The square wave is used to drive a tuned circuit which is tuned to the fundamental frequency of the square wave. The tuned circuit is serially connected to the vertical deflection yoke so that the current in the tuned circuit is the same as the current in the yoke. Thus, the beam position is known from the current waveform. The voltage across the tuned circuit is 90° out of phase with the current of the tuned circuit. Therefore, a circuit is provided for detecting the zero-crossings of this voltage waveform which in turn corresponds to the peak of the current in the deflection yoke. The pulses which produce the intensity modulation of the CRT beam are derived from this signal, so that the position of a spot in a character is constant from scan to scan.

This signal starts a pulse sequence which produces one column of the intensity pattern of the dot matrix representation of the character from the drums. The character on the drum selects a block of memory in the ROM. Each word in the selected block contains one column of the dot matrix representation. The starting pulse fetches successive column words from the ROM, loads them into a shift register, and starts the "vertical oscillator", which shifts the column out of the shift register. The output of this shift register is used to intensify the beam. Thus, columns of the dot matrix are displayed on the sinusoidal scan.

In order to display more dots than are in the ROM and hence to better represent a character, an interpolation technique is used. A character is stored as a skeletal matrix which is contained in adjacent memory addresses of the ROM. The magnetic drum memory contains a code which points to the first address of the read-only memory which contains the first column of the skeleton of the desired character. This first column is read out from the read-only memory to a shift register network where it is subsequently serially read out during a portion of one-half (as the ascending half) of the high-frequency sine wave vertical deflection. At a corresponding point, one cycle time later, the memory address register of the read-only memory is incremented to the next address with the second column of the skeleton of the desired character is read into the same shift register network as previously. The information previously contained in the shift register network being provided to another register forming a part of the interpolator circuitry which provides additional intensification points over that provided by the skeleton for the character matrix which is to be presented on the face of the cathode ray tube. In implementing the invention, five skeletal columns of the character matrix

were stored in the read-only memory with seven possible skeletal points per column.

The interpolator processes the serially obtained information from the shift register to determine whether additional intensifying points are to be provided between the rows of the skeletal matrix and between the columns of the skeletal matrix. The output of the interpolator containing both the skeletal and interpolated intensifying character points, is amplified and provided to the z-axis of the cathode ray tube. In order that the z-axis intensity modulating pulses corresponding to the character to be generated on the face of the cathode ray tube occur at the proper position of the upward and downward trace of the high frequency y-axis modulation, a detector is used to monitor the sine wave. Synchronizing pulses are provided to assure that the intensifying pulses coincide on the face of the cathode ray tube from frame to frame for each character so that there is no perceptible jitter.

A more detailed description of the invention follows to be read in conjunction with the following figures, in which:

FIG. 1 shows the order of displaying the dots using sinusoidal scan for the 7 by 5 skeletal points and its 13 by 9 display points for a character matrix.

FIG. 2 shows display points for simple interpolation.

FIG. 3 shows the effect on the vertex of an oblique angle of a character representation by strong interpolation.

FIG. 4 shows characters using simple and strong interpolation.

FIG. 5 is a block diagram of the character generator.

FIGS. 6A and 6B are digitally controlled switch logic diagrams.

FIG. 7 is a parallel-in to serial-out shift register.

FIG. 8 shows the effect on the column dots of interpolation.

FIG. 9 is an interpolator for implementing rules No. 1 and No. 2.

FIG. 10 is an interpolator for implementing rules No. 1 through No. 4.

FIG. 11 A-D show the effect on a skeletal matrix of rules No. 1 through No. 4.

FIG. 12 is a bit-order-reversing shift register.

FIG. 13 is a block diagram of a shift register network for implementing rules No. 1 through No. 5.

FIG. 14 shows the data present in the interpolator.

FIG. 15 shows undefined values in the window matrix.

FIG. 16 shows the window matrices definition for a primary and secondary scan.

FIG. 17 is the logic diagram of the shift-register network for implementing rules No. 1 through No. 5.

FIG. 18 shows the patterns in the window matrix for strong interpolation.

FIG. 19 is a block diagram of the interpolation network.

FIG. 20 is a logic diagram of the interpolation network.

FIG. 21 shows the blanking amplifier circuit.

FIG. 22 is a block diagram of the phase-locked loop.

FIG. 23 is a frequency diagram of the phase-locked loop.

FIG. 24 is a logic diagram of the horizontal generator.

FIG. 25 is a schematic diagram of the tuned amplifier.

FIG. 26 is a block diagram of the deflection circuitry.
FIG. 27 is a logic diagram of the vertical oscillator.
FIG. 28 shows a portion of the contents of the read-only memory.

FIG. 29 is a block diagram of circuitry of the read-only memory - interpolator interface.

FIG. 30 is a logic diagram of circuitry of the read-only memory.

SEGMENTING A SINUSOIDAL SCAN

A matrix using the sinusoidal scan may be considered to be a sine wave crudely divided into display sections and adjoined curved sections. Where the display region is defined to end and the curved region begins is a subjective question. The display region includes all zero-crossings where the second derivative of the sine wave is zero, and thus it is reasonable straight. This linear display region can be segmented by the character generator into discrete equal intervals of time. Each interval corresponds to one dot of the matrix, and is intensified if that dot is on. Note that line segments, not points, are displayed on the CRT. The beam does not stop moving to display a dot.

One column of a matrix can be displayed in each display region of the sine wave. An m column matrix can be displayed in $m/2$ cycles of the sine wave since one cycle has 2 linear regions. FIG. 1 shows a 13 by 9 display matrix with some of its points numbered in the order in which they will be displayed using the sinusoidal scan.

SINUSOID DISTORTIONS

There are two characteristics of a sine wave which tend to distort the squareness of a matrix displayed in the above manner. First, adjacent display regions do not have the same slope, and thus they are closer together in either the top or the bottom of the scan. Therefore, the horizontal distance between two dots varies in the displayed matrix. Secondly, the sine wave moves more slowly near the top and bottom in its curved sections. The dots tend to get bunched up near the top and bottom of the scan, and spread out in the center, causing the vertical distance between two points to vary. The magnitude of these aberrations are a function of how much of the sine wave is used to plot the matrix.

VERTICAL DISTORTION

The vertical distortion produced by the sine wave scan can be reduced. The solution is to use only the center portion of the sine wave where the beam velocity is relatively constant. The major problem is that it forces the operating frequency of the character generator to be higher. If one plots an m by n matrix in $8.5\mu\text{s}$ using the fraction p of the sine wave to display, then the frequency f_1 at which the dots must be displayed is given by:

$$f_1 = m \cdot n / p \cdot (8.5\mu\text{s})$$

Frequency is inversely proportional to the fraction of time used to scan, p .

Consider displaying a 13 by 9 display matrix. If p is assumed to be $\frac{1}{2}$, f_1 is 20MHz, within the range of ordinary logic. The matrix with $p = \frac{1}{2}$ has negligible distortion and does not perceptibly affect the quality of the displayed character.

SIZE OF MATRIX

The quality of the digitally stored image is strongly dependent on the number of points displayed per character. It is assumed that the information about the shape of each character is stored in a dot matrix. A 7 by 5 dot matrix gives fair quality capital letters, while a 9 by 7 matrix gives good upper and lower case letters. The size of a vertical column (or position in a column) is noted first, followed by the size of the horizontal row (or position in a row). Because complicated Greek letters are desired, better than the 9 by 7 matrix must be generated.

A 9 by 7 matrix has roughly twice as many dots to be stored per character as does a 7 by 5. Increasing the size even further would increase the cost of the digital memory used to store the tables. It would be a costly brute-force solution to go to a larger matrix.

SMOOTHING ALGORITHM

The advantage which one gains when a larger matrix is used seems to be that the corners of the characters are more rounded and oblique lines are less ragged. Being able to finely position the parts of a letter (e.g. the middle bar of an E) does *not* seem to be very critical. The eye is accustomed to recognizing both handwriting, which has inherently variable dimensions, and printing, which is done in many different fonts. Little information is conveyed by the exact placement of a character's parts.

The above observation leads to a method to economize on the size of the character matrix table. If only the basic shape of the character is stored in a small *skeletal* matrix, this matrix, if displayed, would look roughly like the desired characters, but because the matrix is small, the character would have many jagged features. It is possible to add additional dots between the skeletal points to make the letter well rounded. This smoothing occurs as each matrix is being displayed. Thus a much larger matrix can be displayed than is stored in the read-only memory. The memory reduction is especially advantageous in a very large symbol set. The dot matrix for approximately 200 symbols, including upper and lower case Greek and English letters, along with many scientific symbols, is desired to be stored in digital memory.

INTERPOLATION ALGORITHM

The simplest type of expansion from the skeletal matrix is to add a new row (column) between each of the specified rows (columns). FIG. 1 shows the matrix with skeletal points marked with x and displayed points marked with ".". If the stored matrix is n by m , the displayed matrix will be $(2n-1)$ by $(2m-1)$. With large values of n and m , the displayed matrix has roughly four times as many dots as are in the skeletal matrix, leading to a 75 percent saving in memory.

It is necessary to specify a smoothing algorithm which works well in characters and also has an economical realization. The following paragraphs attempt to derive the algorithm.

One recognizes a specific shape in a certain area of a character independently of the shape of the rest of the character. To detect the curved bottoms of the letters "O" and "U" for example, a person does not have to look at the top half of the characters. The smoothing algorithm should therefore be able to generate the

shape of a particular feature of a letter using only the information in a small area of the skeletal matrix. The smoothing algorithm will be given only those skeletal points which are within a certain distance of the point being computed. There is a window centered about each point which contains all the information necessary to specify the value of that point.

The curves and diagonals which need smoothing on a character can occur on any area of the character. Thus the type of smoothing is invariant to translation. The smoothing algorithm should use the same rules for smoothing no matter where in the image the window is placed.

It is possible to form a set of all the patterns which can appear in the window around a given point. A certain subset of this set will always cause the given point to be "on". Thus, the smoothing consists merely of combinational logic driven by the values of the skeletal points inside the window.

SIMPLE INTERPOLATION

Two interpolation algorithms were incorporated in the apparatus: the second is a refinement of the first. The problem of explaining the smoothing is complicated because there are four classes of display points which must be handled differently. They are:

- Type 1: those which coincide with a skeletal point.
- Type 2: those which are one unit above a skeletal point.
- Type 3: those which are one unit right of a skeletal point.
- Type 4: those which are one unit above and to the right of a skeleton point.

A different combinational logic net must be made for each of them. The more simple type of interpolation can now be described in tabular form. The first column of the table below tells the type of point being described by that line, and the second specifies the condition necessary for that point to be on.

SIMPLE INTERPOLATION

Type point — Condition to be on

- Rule 1. (on skeleton)—corresponding skeleton point is on.
- Rule 2. (above skeleton)—both skeleton points above and below are on
- Rule 3. (right of skeleton)—both skeleton points to left and right are on
- Rule 4. (right and above)—two two closest skeletal points on one diagonal are on *and* the two closest skeleton points on the other diagonal are off.

Each of the four interpolation rules above is illustrated in the simple pattern in FIG. 2. The matrices shown in the left column are skeletal matrices. The matrix to the right of each skeletal matrix is the display matrix which was derived from it.

It is the function of the second condition in rule 4 to inhibit diagonal interpolation if there is a "square interpolation" possible. Thus the square corners of characters are not rounded.

STRONG INTERPOLATION

A second set of interpolation rules is illustrated in the simple patterns of FIG. 3. This set of rules contains all the rules previously stated plus one more rule:

STRONG INTERPOLATION

Rule 1 through Rule 4 plus:

Rule 5: — If a skeletal point is part of both a diagonal pattern and a vertical (or horizontal) pattern forming an obtuse angle, then that skeletal point is moved to an adjacent display point as shown in FIG. 3.

The skeletal point mentioned in rule 5 is the point which is at the vertex of the obtuse angle formed by the diagonal pattern and the vertical (or horizontal) pattern. To a certain extent, this point sticks out. The intention of rule 5 is to soften that point's effect by moving it inwards.

Rule 5 helps in many cases, but it also produces the undesirable patterns of lone dots in a row, and square shaped patterns in curved areas. These are evidenced in the characters of FIG. 4. Therefore, the hardware implementation of the smoothing algorithms has provision for choosing either simple or strong interpolation depending on the character to be represented, the KEY WORD in the ROM 53 block of storage containing the character having a code to depict either smooth or strong interpolation.

The character generator is composed of a data section 51 which processes the matrix data, and a control network 52 which directs the data processing. FIG. 5 shows a block diagram of the whole system.

A commercially available "wire-rope" read-only memory (ROM) 53 stores the skeletal matrix for each of the 192 characters in the character set. Upon receipt of an ASCII code from the magnetic drum memory 54, the read-only memory 53 looks up the 7 by 5 skeletal matrix which corresponds to the code received. Details of the lookup process are discussed in the Read-Only Memory section. It sends that matrix to the interpolator 55. The interpolator 55 expands the 7 by 5 matrix to a 13 by 9 matrix, using either the simple or strong interpolation algorithm as preferred for the character being presented. While the matrix is being expanded, it is put in the proper format to be displayed using the sinusoidal scan pattern.

DATA SECTION 51

The intensity pattern in the display matrix must be presented to the cathode ray tube 56 sequentially in order to be displayed using the sinusoidal scan. This serial nature of the output of the interpolator 55 can be exploited to improve its design. To better understand this relation, a series of examples which lead to the complete interpolator will be discussed.

SIMPLE "FORMATTOR"

First consider the simple problem of displaying an existing m by n matrix without interpolation. Suppose the matrix is divided up into n m -bit words which are, presented to the formattor one word at a time. To simplify matters, assume for the moment that only every other display region is being used to display dots. In this case all scans used for displaying are in the same direction. For purposes of discussion this scan direction will be considered *down* the columns of the matrix.

The formattor described here is nothing more than a parallel to serial converter. The converter accepts one m -bit word of the matrix at a time. It converts that word to a serial string before accepting the next word. Two possible implementations exist. The first is an m -

position digitally controlled switch for purpose of illustration, $m=4$. FIG. 6A shows a 4-position switch 63 and FIG. 6B shows the switch symbolically. When the control bits 64 form a binary k , the k 'th input is presented at the output. The control bits are generated by a 4 state counter 61, sequenced from 0 to $m-1$ by a clock 62 which occurs whenever another bit of the serial output is desired. The second conversion method uses a parallel-in, serial-out shift-register 71. FIG. 7 shows a four-bit shift-register converter 71. First, the information B_0-B_3 is transferred in parallel into the m bits of the register 71. Then the m bits are serially shifted out of the register by the output clock 12. The "switching" method is less complex, but the shift-register method has the advantage that it can be used as a memory unit also. Both methods will be used in the final interpolator 55.

COLUMN INTERPOLATION

Consider augmenting the formatter to do a simple type of interpolation described in interpolation rules no. 1 and 2. A new dot will be added by the interpolator directly after each dot of the row being formatted. Two values are generated on one output line 92 for each bit B_k . They are presented one after another, one at time k by the primary dot pulses 93 and the other at time $k + \frac{1}{2}$ by the secondary dot pulses 94. These rules cause "display points" 81 to be added if they are between two "skeletal points" 82. FIG. 8 shows a matrix under this interpolation.

Although this interpolator could be implemented using two digitally controlled switches 63, it is much simpler to design it using flip-flops 95 connected as a shift-register converter 90 as shown in FIG. 9. The flip-flops 95 are set to the inputs B_0-B_4 through their "set direct" inputs 96. The converter 90 presents the last bit B_0 of the shift-register 91 which the signal PRIMARY DOT 93 is true, and it presents the logical AND of the last bit and the next to the last bit when the signal SECONDARY DOT 94 is true. The shift register is shifted on the trailing edge of SECONDARY DOT pulse 94.

SIMPLE INTERPOLATION

Interpolation rules no. 1 and no. 2 have been effectively implemented. Additional problems are encountered in adding rules no. 3 and no. 4, which allow for adding dots in horizontal or diagonal rows. The extra dots generated by these rules all fall into an extra column which must be added between the columns of the skeletal matrix. The interpolator must generate this "secondary row" after it generates each "primary row" described by the first type of interpolator. Another problem is that these rules involve the comparison of points in two adjacent rows. Both rows must be present in the interpolator at the same time.

One way to have both rows present is to add another m -bit shift register as shown in FIG. 10. This register is used to hold the row previously processed. When the interpolator is generating a primary row, the input B_k 's are converted to signals on the lines W_{11} and W_{21} . They are used as before to perform type no. 1 and no. 2 interpolation. While this interpolation is being performed, shift register 2 is being loaded with the serial output of shift register 1.

When the interpolator is generating a secondary column, a new word of B_k 's are converted to signals on lines W_{11} and W_{21} . In addition, W_{12} and W_{22} scan the

values of the previous column in the same manner that W_{11} and W_{12} do for the new columns.

The outputs W_{11} , W_{12} , W_{21} , W_{22} are named with subscripts because they can be thought of as a matrix. This matrix is actually the window matrix centered about the dot being generated. The shift register structure described in FIG. 10 causes the window to slide down the rows of the matrix covering two columns at a time. At any instant, the window contains all the information necessary to do the desired interpolation. The values in the window, along with the type of column (primary or secondary) and type of point (primary or secondary) are fed to a combinational logic net to determine whether that point is on.

An example may be helpful at this time. FIG. 11(a) shows a skeletal matrix, and FIG. 11(b) shows the corresponding interpolated matrix. FIG. 11(c) shows the contents of the two shift registers as the first two columns of the interpolated matrix in FIG. 11(b) are being generated.

The first entry in FIG. 11(c) labeled "example", shows the two shift registers containing the arbitrary variables $A, B, \dots, J$. The last two outputs of SR1, (D and E) and SR2 (I and J) are used to form the window matrix. The purpose of the literals is to show the relation between the bits of the shift registers and the bits of the window matrix. In the columns labeled "Primary" and "Secondary", the literals $A, B, \dots, J$ have been replaced by specific values 0, 1, or ϕ . The symbol " ϕ " represents the don't care condition; the interpolation is independent of its value.

Before output time 1.0, the binary word 01110 (the first column of FIG. 11(a)) is entered into SR1. As the primary scan progresses, this word is shifted into SR2. In the process, consecutive pairs of bits in the word are present in the window matrix.

FIG. 11(d) shows the patterns which must be present in the window matrix if the output value is to be a 1. A heavy dot indicates that the bit must be a 1, a circle indicates that the bit must be a 0, and a light dot indicates that the value of that bit has no effect on the decision. Given that a certain type of dot (primary or secondary) is to be generated in a certain type of column (primary or secondary), one can look in the appropriate portion of FIG. 11(d). If there is a pattern in that portion which matches the pattern in the window matrix, then the output of the interpolator is a 1. In this manner the binary string 0011111100 is produced by the primary scan. This is the first column in FIG. 11(b). The last column in FIG. 11(c) shows which rules were used to generate each 1 in the string.

After the primary scan, a new word (binary 10010) is loaded into SR1. This is the second column of FIG. 11(a). Another scan is performed except this time the rules for interpolation are for a secondary row. The string produced, 001000010, is the second column of FIG. 11(b).

ORDER REVERSING

It is possible to remove the restriction that only scans down a column are used for displaying dots. Dots can be displayed in every display region, whether the beam is scanning upward or downward. The problem encountered is that the display matrix points must be generated in reverse order when the beam is scanning upward. Since the interpolation rules operate equally well on matrices whose columns are upside down, the col-

umn can be generated in everse order, by sliding down a skeletal matrix whose columns have been turned upside down. Thus, all one needs to do is to insure that the skeletal matrix is upside down when generating an upward column. The interpolation can then be carried out in the same manner as on a downward column.

It is necessary to have the matrix in its upside down form every odd scan, and in its right side up form every even scan. The contents of the shift registers must therefore have their order reversed between each scan. Bit 1 is swapped with Bit n , Bit 2 with Bit $n-1$, etc. This feat can be accomplished in one parallel operation in a bit-reversing shift register 120 as shown in FIG. 12. When shifting, information is entered into each cell or flip-flop 121 through its input A. When flipping, information is entered through its input B. (An additional parallel entry path for the B_k 's has not been shown because it is not needed in the next section.) Register 120 is constructed of two Fairchild 9300 circuits 122 interconnected as shown in FIG. 12.

An alternative approach to the problem is to have bi-directional shift registers. The registers would shift forward during one scan, and backward during the next scan. Bi-directional shift registers and order flipping shift registers in themselves require about the same logic to implement. Formation of the window matrix using bi-directional shift registers is more complicated because different bits of the shift register must be used when shifting in different directions. They were not employed for that reason.

STRONG INTERPOLATION

This is the final example in the series of machines: this is the description of the machine which was built. Interpolation rules no. 1 through no. 5, which do all the previous filtering and, in addition, move the vertex of an obtuse angle inwards, can now be implemented. These rules require a 3 by 3 window matrix 131 for generation of the primary columns, and a 3 by 2 matrix for the generation of secondary columns.

It is necessary to form a configuration of shift registers which will produce the larger window matrix 131. The goal is basically an extension of the 2 by 2 configuration. To obtain an extra column in the window matrix, another shift register is added. To obtain an extra row, one clocked delay is added and one more output is taken from each shift register. In addition, part of SR1 has been replaced by a digitally controlled switch to reduce logic. The shift register network 130 which includes SR network 58 for implementing rules No. 1 through No. 5, is shown in FIG. 13, and is described in the following paragraphs.

During the primary scan, one m -bit word is presented in parallel on lines B_k . FIG. 14 showing the data present in the interpolator defines the process explicitly. If the column of the display matrix corresponding to column i in the skeletal matrix is about to be generated, then column $i+1$ is present on the B_k 's at this time. This advance allows the interpolator to look one column ahead of the column it is generating. In the bit reversing shift register 120, SR2 132 is assumed to contain column i , and SR3 133 is assumed to contain column $i-1$. These assumptions will be justified shortly.

Before the scan is started, B_1 is loaded into one of the one bit delays 134 shown in FIG. 13. Before shifting is started W_{21} , W_{22} , and W_{23} of FIG. 16(b) are the first bits in column $i+1$, i , and $i-1$ respectively of the skele-

tal matrix. W_{31} , W_{32} and W_{33} contain the second bit of their respective columns. W_{11} , W_{12} , and W_{13} contain incorrect values. This error corresponds to the situation depicted in FIG. 15. Part of the large window matrix 131 falls outside the skeletal matrix into an undefined area. To prevent any points from being incorrectly added to the interpolator's output, these values are defined to be zero during the first shift state. The AND gates 135 in FIG. 13 perform this function when energized by signals on line 145, 146 from vertical row counter 140.

During the primary scan, the data switches 138 are set by the signal FORWARD (FW) on line 171 so that SR2 132 is shifted into SR3 133 and the B_k 's are converted to a serial string by switch 139 energized by pulses from the vertical counter 140 on line 148, and shifted into SR2 132. The vertical oscillator 141 is controlled by the signal on hold oscillator line 142 provided by start-stop logic 143. Logic 143 provides a start oscillator signal on line 142 in response to the TRACE signal on line 144 and a stop oscillator signal on lines 142-147 when provided by a signal from vertical counter 140. The clocked delay 136 is always driven by the output of SR3 133. It is used to generate W_{33} . Once the registers have been shifted *once*, all the windows contain correct values. For example, W_{31} contains bit 1 of column $i+1$, W_{21} contains bit 2 of that column, and W_{11} contains bit 3. After each shift-pulse 137 from vertical oscillator 141, the bits advance in the shift registers 132, 133 and the window appears to slide down the columns of the skeletal matrix. Until the last shift pulse, all the values of the window matrix are valid. Then part of the window matrix extends below the skeletal matrix. The remedy is again to AND out the invalid levels during the last shift time.

After the primary scan is completed, SR2 contains column $i+1$, and SR3 contains word i . Now the interpolator is ready to generate a secondary column between column i and column $i+1$. Since this column is displayed on the upswing of the scan, it must be generated in the opposite direction. The shift registers must have their order reversed at this time.

During the secondary scan, the data flow switches 138 are set by the FLIP signal BACKWARDS (BK) on line 172 to cause the shift-registers 132, 133 to recirculate upon themselves. The undefined conditions caused by the window matrix extending outside the skeletal matrix exist in this scan as before and are treated using the same logic.

Because only a 3 by 2 window matrix is needed to do this secondary interpolation, only two shift-registers, SR2 and SR3, take an active part in producing the window matrix at this time. The digitally controlled switch 139 is not used during this scan. The 3 by 2 window matrix is shown in FIG. 16(a). Unfortunately, the indices of the W 's do not coincide with their position in the window matrix. This is only a conceptual problem — the combinational logic can be wired around it.

After the secondary scan is completed, SR2 136 contains word $i+1$, and SR3 133 contains word i . It is now necessary to re-flip the bits of the shift-registers back to the forward order, since the next scan is in the forward direction. The net effect of advancing and flipping the columns during the primary scan, and recirculating and flipping them during the secondary scan is to put SR2 into SR3 and to put the B_k 's into SR2, both in unreversed form. It is easy to see that after a few cycles

SR2 will always contain the previous column processed, and SR3 will contain the column processed before that. Thus, our original assumption to that effect was justified.

FIG. 17 shows the hardware logic diagram of the shift register network 58 which performs all the shifting and flipping operations necessary to generate the window matrix.

FIG. 18 shows the patterns which must appear in the window to generate the four types of points in the display matrix for strong interpolation. The combinational logic of block diagram 59 of FIG. 19 and its corresponding hardware logic diagram of FIG. 20 implements these rules.

The window matrix inputs W_{11} - W_{33} are applied as inputs to the logical blocks Primary Dot, Secondary Column 191, Primary Dot, Primary Column 192; Secondary Dot, Secondary Column 193; and Secondary Dot, Secondary Column 194. The output of blocks 191-194 are gated with the Forward and Backward signals on lines 171, 172 respectively, in the "type of column" logic block 195 to provide signals to the primary/secondary dot FF 196 and the secondary dot FF 197. The primary dot on line 198 and the secondary dot on line 199 are derived from the leading edge circuit 200 and the trailing edge circuit 201 operating on the shift pulse on line 137 after being delayed in unit 202.

Two additional features are shown in FIG. 20. It is possible to either apply rule no. 5 or not. If the level labeled FILTER MODE 1 on line 204 is TRUE, rules no. 1 through no. 5 are used to generate the display matrix. (Strong interpolation is performed.) If FILTER MODE 1 is FALSE, only rules no. 1 through no. 4 are used. (Simple interpolation is performed.) This provision was added because best results are obtained by interpolating some characters using simple interpolation, and other using strong interpolation. An extra bit is stored in the "key word" of the read-only memory for each character, to tell which type of interpolation is to be applied.

The second addition was required because of the high operating frequency of the interpolator. Since a dot is generated every 25 nanoseconds, races in the combinational logic become critical. Note that strings of functionally useless inverters or R-C delays have been added to several paths of FIG. 20. This forces all paths to be within about one gate delay of each other. To prevent any remaining races from reaching the CRT 56, the two outputs 198, 199 for each shift-pulse are stored in two buffer flip-flops 196, 197 before they are multiplexed onto one output line 205.

The blanking amplifier 57 is shown in FIG. 21. It is capable of driving a 30 volt waveform into 20pF load with a rise time of 6 ns on line 206. The output stage is a complementary common emitter amplifier.

TIMING CHAIN

The timing chain directs the operation of the data section of the interpolator just described. Supplied with only a single pulse every character, the timing chain forms some 200-odd time periods which are necessary to display that character. Design was complicated by the following facts:

1. The timing of the unit must follow the timing of the drum 54. The unit cannot operate asynchronously.

2. A periodic signal must be used to produce the sinusoidal scan, and the appropriate frequency is not recorded on the drum.

3. Delays in the deflection circuits must be compensated for by the timing circuitry.

It proved impossible to synchronize all timing signals with one high speed clock. Pulses are consequently formed by subdividing the pulses from the drum into many shorter ones. The horizontal generator produces a fixed number of periodic column scans for each character pulse from the drum. One character-time interval is thus divided into the correct number of column intervals. Each of these display regions is then segmented by the "vertical generator" into dot-time intervals.

HORIZONTAL TIMING

It is necessary to fix the number of columns generated per complete character. When displaying a full screen of characters, it is necessary to leave some space between characters to prevent them from running together. One solution is to add a staircase signal to the ramp before it goes into the horizontal deflection amplifier. The staircase would advance to a new step after each character and move the scan pattern right for the next character. Unfortunately, this scheme requires a settling time of approximately one half the sine wave period and is therefore impossible with a 10 μ s deflection system.

Another more tractable method of spacing is to keep the constant sinusoidal scan across the line and leave blank scans between characters. It is desirable to use an even number of scans to plot a character and its space, since then an integral number of sine wave periods would be used, and all characters would begin on the same phase of the sine wave. Because there are an odd number of scans, nine, consumed in plotting the 13 by 9 matrix, there must also be an odd number of blank scans. One scan gives insufficient space, but three blank scans yield the proper spacing. It is undesirable to go to five or more blank scans, because the percent of time used for plotting is decreased, and hence frequencies in the interpolator must go up. With three blank scans, a total of 12 scans, or six sine wave periods, are necessary per character for a 9 by 7 matrix.

It is therefore necessary to generate six cycles of a sine wave per character by a phase locked loop oscillator 510.

PHASE-LOCKED LOOP

FIG. 22 shows the details of the phase-locked loop 510 shown in FIG. 5. It generates a frequency 6 times f_o at output 223 from a reference signal at point 221 of f_o , where f_o is the frequency of the characters on drum 54. A voltage-controlled oscillator runs at about $6f_o$. The output is divided by 6 in counter 228 to produce a signal at point 222 of about f_o . This signal is compared to the reference frequency at point 221 in counter 224 and if it is not the same, an error signal is sent to the integrator 225, filtered in filter 226, and the frequency of the voltage-controlled oscillator 227 is changed appropriately.

It is the phase of the reference signal at 221 that is compared with the phase of the generated signal at point 222. The comparison is done by a 4-state up-down counter 224 which is incremented at every pulse from 221, and decremented by every pulse from 222. The counter stays in its highest state, called state 3, if

it is incremented while in that state—it does not go to its lowest state as a modulo 4 counter would do. It also stays in the lowest state, called state 0, if decremented in state 0. If the counter were started in state 1, the number in the counter at any subsequent time would measure the difference in the number of pulses at point 221 from the number at point 232, provided this difference never exceeds -1 or $+2$.

The output of the counter 224 is an analog -1 unit if the counter is in state 0 or 1, and is an analog $+1$ unit if the counter is in state 2 or 3. Therefore the integrator's 225 output has a negative slope if the counter contains a 0 or 1 and has a positive slope if the counter contains a 2 or 3. The output of the integrator is fed to a low pass filter 226, which rolls off with two poles ($Q=5$) at about $f_o/10$. The filter reduces the pulse to pulse frequency modulation of the oscillator.

There are two distinct modes of operation of the loop, each characterized by the counter being in a different set of states. First is the capturing mode. If the synthesized frequency at point 22 is much larger (smaller) than the reference frequency f_o at 221, then the counter will spend all of its time in states 0 and 1 (states 2 and 3). The integrator would therefore decrease (increase) its value, and the oscillator will also decrease (increase) its frequency. Finally, the frequency at 222 will actually be a little smaller (larger) than that of 221. Two pulses at 221 (222) will occur between pulses at 222 (221), and the loop enters state 2 (state 1) from state 1 (state 2).

The second mode is called the locked mode, and is characterized by the counter being in state 1 or 2. To show that the loop does indeed lock when in states 1 or 2, it is necessary to get the open loop frequency response of the system. FIG. 23 shows the frequency response of the various parts of the system. Two of the boxes require some special explanation. The first is the comparator. When operating exclusively in states 1 and 2, pulses come to the comparator alternately from points 221 and then from 222. One pulse from 221 takes the comparator from state 1 to state 2, and then a pulse from 222 takes it from state 2 to state 1. The time average of the output of the counter, taken over one of these cycles, is zero if the two signals are exactly 180° out of phase. The time average, taken in the same way, is proportional to the difference of the relative phases from 180° . In terms of frequency of the square wave input, the output is proportional to the integral of the difference of frequencies. Therefore, the comparator integrates its input frequency to obtain its output, and is denoted as an integrator.

The other integrator in the system also contains a zero at about 100 Hz. The purpose of the zero is to reduce the phase shift when the loop gain is unity, thus making the loop stable.

The loops was designed to have unity gain at about 1 kHz. This low crossover frequency is tolerable since reference frequency cannot change very fast. The reference signal is derived from the drum which takes seconds to change its frequency because of its large moment of inertia. The loop also has the advantage of smoothing out any jitter in the drum signals caused by incorrect recording or domain granularities.

The hardware realization of the phase locked loop, 510, along with some circuits to be discussed, is shown in FIG. 24 as the horizontal generator 511.

VERTICAL DEFLECTION CIRCUITS

The output of the phase locked loop 223 is low-pass filtered into a sine wave at 223, amplified, and sent to a tuned transformer 251 in the amplifier 250 of FIG. 25. The output of the transformer 251 is added to the vertical deflection amplifier 261 of FIG. 26 by placing the output in series with the deflection amplifier 261. The sum is sent directly to the y vertical coils 512 of the display CRT 56. This vertical sum signal along with the horizontal ramp current of the horizontal amplifier 262 in the x horizontal coils 513 produce the sinusoidal scan pattern for the CRT 56.

Although the sinusoidal scan does bypass the settling time of the vertical system, it does not allow accurate positioning of the beam. This is an inherent problem with the sinusoidal scan which becomes very important at this point — the phase delay of the vertical sinusoid amplifier is not constant. The main problem is that the sine wave must pass through a high Q transformer 251 before it is added to the vertical raster signal. Small frequency shifts in a high Q system about its resonant frequency produce large frequency shifts. Both the input frequency and the resonant frequency vary. The scan frequency varies because the timing for the whole system is ultimately derived from a clock track recorded on the drum. Jitter on the clock tracks or hunting of the drum 54 causes the frequency of the sine wave to change. The resonant frequency of the amplifier changes because the inductance of the vertical coils 512 on a magnetic yoke is in parallel with the tuned transformer. Since the yoke is non-linear, its inductance, and hence the resonant frequency of the transformer 251 change depending on the position of the character on the screen. Changes of the scan frequencies about the resonant frequency of the toroidal transformer 251 cause a large phase shift. In addition the frequency response of the amplifier can not be altogether stable, because of temperature drifts and stray capacitance. These arguments imply the position of the beam can not faithfully follow the sine wave used to drive it.

Although it is not possible to predict the position of the beam knowing only the input to the amplifier, it is possible to sense the beam position by measuring the yoke voltage. To derive this relation, it is recalled that (1) the angle of deflection in a CRT is proportional to the B-field in the yoke; (2) the B-field is proportional to the yoke current; and (3) the change in current in the yoke is proportional to the voltage across the yoke. These statements together imply that the voltage across the yoke is proportional to the velocity of the beam.

The voltage across the yoke is therefore fed to a differential amplifier detector 241 and clipped about its zero crossings. The output of the detector 241 (called FW which stands for a forward direction of scan) becomes ONE when the beam is at its maximum height, and stays ONE while the beam is scanning downward. FW becomes zero when the beam is at its lowest point, and stays ZERO while the beam is scanning upward. The output BW of detector 241 is the complement of FW.

All the internal timing signals of the interpolator-formatter 55 are derived from the FW signal, because it is the only accurate measure of the beam's position. The phase-locked loop output 223, which drives the beam, cannot be used because it jitters with respect to FW.

FW itself is used to tell the direction of the scan, the hence whether the scan is a primary or secondary column. The falling edge of FW is used to gate the state of the divide-by-six counter 228 in the phase-locked loop 510 into a three-bit latch register 242. This register 242 contains the number of the current column being generated. A latch had to be used to force the column count to change synchronously to the change of FW.

A signal called TRACE is formed from FW at the output 243 of differentiator 244. TRACE has a pluse whenever FW changes from ONE to ZERO or from ZERO to ONE. TRACE is used to order-flip the shift-registers, since this must happen before each scan. If TRACE is delayed by the appropriate amount in delay circuit 514, it occurs at the beginning of the region used to display the matrix. The delayed TRACE pulse 144 starts an oscillator 141 which starts generating SHIFT pulses 137.

VERTICAL TIMING

The SHIFT pulses 137 are generated by a 20MHz oscillator 141 shown in FIG. 27. SHIFT pulses do three things. Primarily, they are used to shift the shift registers of FIG. 17 which generate the window matrix. Secondly, they are used to reclock the output of the combinational logic of FIG. 20 which does the interpolation. On the leading edge of SHIFT, the primary dot is outputted to the z-axis of the CRT. On the falling edge the secondary dot is outputted. The third task of the SHIFT pulses is to drive an m -state counter 140 of FIG. 17 which tells what bit in the current column is being generated. When this counter 140 is zero or $m-1$, one of two levels is sent to the window generator 135 to signal that certain bits of the window matrix are invalid. The counter 140 also turns off the oscillator 144 after m shift pulses have been generated. The above process causes m SHIFT pulses to occur after each delayed TRACE pulse. If the frequency of the oscillator 144 is right, the m 'th pulse will occur exactly at the end of the display region.

READ-ONLY MEMORY

The previous section described the hardware necessary to transform a skeletal matrix into a display matrix, and to plot that matrix of the CRT. It was assumed throughout those sections that the interpolator was somehow supplied with the correct skeletal matrix. This section describes the process which supplies the interpolator with the appropriate skeletal matrix.

STORING SKELETAL MATRICES

The organization of the skeletal matrices in the ROM 53 is a compromise between two extremes. One extreme is to have 35 bits per word in the memory, and thus one whole 5 by 7 matrix would fit into one word of the memory. This requires handling 35 bits of information in parallel somewhere in the machine. Duplicating data paths 35 times, even if for only one register, is an unpleasant task. A more serial approach seems better. The other extreme in organization is totally serial. Consider storing only one bit per word. Thus 35 words are necessary to store one skeletal matrix. Unfortunately, this method requires 35 cycles of the memory to retrieve each character, and forces the memory to operate at unrealistic speed.

It was decided to store each matrix in five words, with each word containing seven bits. Each word is one column of the matrix, and thus the memory scheme is compatible with the interpolation format. This organization also eliminates the problems associated with the totally serial and the totally parallel structures of the previous paragraph.

MODIFIED ASCII CODE

All information which is to be displayed on the CRT screen is stored in ASCII-coded form on the magnetic drum 54. Characters are displayed on the CRT screen in the same order that they are recorded on the drum track. As one character is read serially from the drum, bit by bit, it is converted to parallel in a serial-to-parallel converter 263, and used to drive the character generator address map 291.

The standard ASCII codes have 10 bits. There are seven bits of coded information, and three bits of header. The header is the same for all characters, so that there are 2^7 or 128 possible codes. The 128 codes are divided into two groups: control codes and visible codes. Visible codes include all letters, numbers, and punctuation marks. Control codes include carriage return, line feed, tab, status, etc., and are characterized by the fact that bit 6 and bit 7 of their code are both zero. Therefore, there are 32 control codes, and 96 visible codes in the ASCII character set.

One design objective for the display system is to have a 192-character set. For this reason, a character set flip-flop 292 is included to double the number of visible codes. Depending on the state of this flip-flop, characters are selected from one of two possible character sets. Two control codes in the standard ASCII set are reversed to set and reset the character-set flip-flop. Other control codes operate independently of the character set, since functions like line feed and carriage return are needed in both sets. Control codes received by the Address Map 291 are interpreted as spaces.

The state of this flip-flop 292 is sent to the character generator, along with the seven bit ASCII code. Together they form an eight-bit modified ASCII code. This code has the property that bit 6 and/or bit 7 is always ONE. The character generator ignores codes in which bit 6 and bit 7 are both ZERO: in that case it displays a blank or space.

ADDRESS MAP

It is necessary to store 192 5-word skeletal matrices in the ROM so that the blocks are easily retrievable. Most memory is produced in units which contain 2^n words, where n is an integer. The problem now becomes that of packing these 960 words into 2^{10} or 1,024 words of ROM memory so that access is easy.

The easy access map is motivated by the following factorization of the numbers involved. There are $15 \cdot 2^8$ words to be stored in $16 \cdot 2^8$ locations. Therefore, store 15 words, or three skeletal matrices, in each block of 16 locations. Since bits 6 and 7 of the ASCII code together can assume only the three values 01, 10, and 11, (the cannot both be zero), they are used to tell which of the three skeletal matrices in the block of 16 words is desired. Then bits 1 through 5 and bit 8 (the character-set flip-flop 292) tell which of the 2^8 possible blocks of 16 locations is being considered.

The columns of the skeletal matrices are stored in blocks of five consecutive words. If bit 6 is ZERO and

bit 7 is ONE, locations 0 through 4 in the appropriate 16 word block are addressed. If bit 6 is ONE and bit 7 is ZERO, locations 5 through 9 in the 16 word block are addressed. Finally, if both bits 6 and 7 are ONE, locations 10 through 14 are addressed. Location 15 of the block is not used presently, although a use for it will be described in the Key Word section.

To implement the above in hardware, a 16-state counter is used, the Memory Address Register 293. The counter 294 is preset to either 0, 5 or 10, depending on the values of bit 6 and bit 7. The counter is then indexed by a pulse from timing unit 285 to obtain the next column in the skeletal matrix. FIG. 28 shows the contents of the first 32 locations in the read-only memory.

This mapping has one pleasing feature: it is simple. It requires only three NAND gates 295 and one four-bit counter 294. For implementation, it does not require that the whole memory address register 293 of the ROM be a counter. A portion may be a latch 293. All columns for any given character are in the same block of 16 words. Only four bits of the memory address must count.

KEY WORD

The sixteenth word in each block is used in the following manner. The bits of this word can be divided amongst the three skeletal matrices stored in the block. Two tag bits per matrix can be stored in this last word thus using 6 of its 7 bits. The first tag tells what type of interpolation is to be applied to the matrix, simple or strong. The second tag could be used to tell the vertical position at which the matrix should be plotted on the CRT. If the character should extend below the line (e.g., *g,j,p,q,y*), the 7 by 5 matrix is lowered to make the character extend below the line. This lowering could be achieved in the timing chain by delaying one edge of FW (see section on Vertical Deflection Circuits) for regular characters, and the other edge of FW for low characters. Thus the display matrix could be shifted downward for low characters.

This second tag was not used in the prototype character generator. Since the ROM memory 53 which was purchased had eight bits per word, and since the interpolator was initially constructed for an eight-bit column, an 8 by 5 skeletal matrix was used. The bottom row of the matrix is always zero for regular characters, and the top row is always zero for low characters. Thus the non-zero portion of each character is actually contained in a 7 by 5 matrix.

It is necessary to retrieve the key word by the key word gates 295 for a character before that character is plotted to determine whether simple or strong interpolation is to be used and to store this information in the interpolator flip-flop 297. As explained in the section on Horizontal Timing, six cycles of the sinusoidal scan are generated on the CRT to plot one character. Only five of these sinusoids are actually used to plot points; one scan is always blank. A read-only memory cycle must precede each of these five plotting cycles to retrieve the new column which the interpolator needs to begin that scan. During the sixth cycle of the sinusoid, the interpolator needs no new data. The read-only memory fetches the key word for the next code to be displayed at that time.

The interface for the read-only memory 53 is shown in FIG. 30. It contains the address map 291, the mem-

ory address register 293/counter 294, and the timing 295 necessary to make the memory operate in the manner described. The counter 294 advances to the next address of the character in the ROM 56 by pulses from the timing unit 295. The space flip-flop 298 causes the control code in the ASCII character set to produce a SPACE and it also provides a space between characters.

DEFLECTION CIRCUITRY

The output of drum 54 provides, through serial-to-parallel converter 262, a frame synch word which is decoded by frame synch character decoder 266 to provide reset pulses to column counter 264 and row counter 265 of the deflection circuit of FIG. 26. Converter 262 also provides a pulse for each character which is provided to column counter 264. Counter 264 is designed to count from 0 to 63 before recycling. During the count of 56-63 corresponding to columns 56-63, the counter 264 provides a reset pulse to integrator 267 which increases its output linearly during counts 0-55 to provide the input to horizontal drive amplifier 262. At each count of 64, the column counter 264 provides a pulse input to row counter 265. The output of row counter 265 is applied to digital-to-analog converter 268 thus providing a staircase input to vertical deflection amplifier 261. Row counter 265 is reset at a row count of 31 during which time the vertical retrace occurs.

HORIZONTAL DOUBLE FREQUENCY

The sinusoidal deflection as stated earlier tends to distort the squareness of the displayed matrix. One form of distortion is that the vertical display regions to not have the same slope, and thus they are closer together at either the top or the bottom of the scan. Although the apparatus of this invention produced acceptable characters without correcting this distortion because it was small for the value of $p = \frac{2}{3}$ used (see the section on Vertical Distortion), a larger value of p , resulting in a lower operating frequency for dot generation or a greater number of dots for the same frequency, might be used without increased distortion if the distortion could be reduced. The solution is to change the scan pattern slightly by superimposing another sine wave onto the horizontal ramp signal. If the horizontal sine wave is twice the frequency of the vertical sine wave, and it has the correct phase and amplitude, the scan pattern will have display regions which are almost vertical. Apparatus for accomplishing this function is shown in FIG. 5 where the TRACE signal is amplified and filtered in unit 515 which drives the tuned transformer 516 in series with the horizontal deflection amplifier 262.

VARIABLE DOT FREQUENCY

The vertical distortion, caused by the bunching of dots, near the top and the bottom of the sine wave, can be corrected by changing the frequency of the oscillator 141 which generates the dots. The horizontal sine wave can be used to frequency modulate oscillator 141, causing it to oscillate faster in the center of the sine wave, and slower near the tops and bottoms of the sine wave. Thus the dots appear to be more equally spaced in the vertical direction.

PHASE-LOCKED LOOP

The drum used in the system did not have the sinusoidal scan frequency recorded on it because it was already in existence at the time the apparatus described here was reduced to practice. It was therefore necessary to resort to the phase-locked loop to generate six cycles per character.

The phase-locked loop could be eliminated from the character generator entirely if frequency for the sinusoidal scan is stored on the drum with the data. The storage could be implemented as a separate track on the drums; or by making the data word on the drum 6 or 12 bits long and using the bit clock from the drum to derive the vertical scan frequency.

ALTERNATIVE READ-ONLY MEMORY

Although the invention has been described using a 7 by 5 skeletal matrix stored in a rope-core read-only memory together with interpolation to expand the matrix to a 13 by 9 matrix because it was more economical to obtain a 13 by 9 matrix by this interpolation technique than by storing the 13 by 9 matrix in the rope-core memory, the substitution of metal oxide semiconductor (MOS) read-only memories of comparable total capacity is apparent when the price and availability of MOS memories permits their use in which case interpolation would not be necessary.

ALTERNATIVE TO MAGNETIC DRUM UNIT

Although the invention was implemented using a drum to store the characters to be displayed, other forms of storage could be used. In particular, magnetostrictive delay lines, or MOS shift registers are both capable of replacing the drum.

CONCLUSION

The preceding description of the character generator when read in conjunction with the figures is considered an adequate description for one skilled in the art to practice this invention. The figures provide details of construction which are not necessary to be discussed for an understanding of the invention but which is considered helpful to those skilled in the art to construct apparatus according to the invention. The detailed logic diagrams show their interconnections with each other by letters or names through which signal paths may be traced in detail.

What is claimed is:

1. A character generator for providing selected characters at desired locations on the face of a cathode ray tube comprising:

means for providing a digital code representation of a selected character,

means for storing said character digital representation at a display address in a first memory in response to said character selection means,

a cathode ray tube having a scanning beam,

means for providing a raster scan having a sinusoidal vertical deflection to said tube,

means for obtaining a stored character code from first memory when its display address corresponds with the position of the raster beam,

a second means for storing in binary form a different matrix representation of each character corresponding to each character code,

means for selectively reading out the character matrix representations as serial binary signals from said character matrix storage means in response to said character codes from said first memory,

means for intensity modulating said sinusoidal deflection in its linear region in synchronism with said serial binary signal to provide a dot matrix representation of the selected character in response to the character matrix representation signal.

2. The apparatus of claim 1 comprising in addition: means for providing synchronizing pulses in response to said sinusoidal deflection,

said means for obtaining said character matrix being connected to said synchronizing pulse means to provide the character matrix at a predetermined position of said raster scan and said sinusoidal deflection in response to said synchronizing pulses,

said means for intensity modulating said sinusoidal deflection being connected to said synchronizing pulse means to provide a series of intensifying pulses corresponding in number of the number of rows in a column of the matrix, said initial intensifying pulse of the series being responsive to said synchronizing pulses to occur near the zero-crossing region of the sinusoidal deflection.

3. A character generator for providing selected characters at desired locations on the face of a cathode ray tube comprising:

means for time sequentially recovering character codes from storage, each code representative of the characters to be generated on the cathode ray tube,

means for applying a raster scan to said tube in response to and synchronized to pulses generated by the time sequence of character codes,

the time of occurrence of each character code corresponding to its desired location on the raster,

means for applying a large sinusoidal vertical deflection to the raster scan, said sinusoidal scan being responsive to said synchronizing pulses,

means for storing a matrix for each character in a time-sequential binary form,

means for providing each recovered character to said matrix storage to cause said matrix storage to provide a dot matrix for the character code in the form of a time sequential binary pulse stream beginning at a time corresponding to the desired location of the character on the raster scan,

means for providing said matrix as intensification pulses to said cathode ray tube in the linear region of the sinusoidal scan in response to said synchronizing pulses, whereby said character is represented in its dot matrix form at the desired location on the cathode ray tube.

4. A character generator for providing selected characters at desired locations on the face of a cathode-ray tube comprising:

a first memory for storing a plurality of character codes of a character set in serial form in the time sequence that each character is to be represented on the face of the tube,

a second memory having a unique character dot matrix stored in binary form for each character of the character set in the first memory,

means for providing a vertical scan frequency corresponding to a multiple of the average frequency of occurrence of character codes.

means for providing a raster scan to said tube,
 means for providing said vertical scan frequency as
 a vertical deflection to said raster scan,
 means for sensing said vertical deflection to provide
 timing pulses, 5
 means for synchronizing said raster scan to said tim-
 ing pulses,
 means for sequentially providing each stored charac-
 ter code from said first memory to said second
 memory, the second memory providing a serial binary 10
 signal in response to each character code,
 means for providing intensifying pulses to the cath-
 ode-ray tube of a frequency higher than that of the
 vertical scan frequency and synchronized with said
 timing pulses in response to said serial binary signal 15
 to form the dot matrix of each stored character
 code,
 said character matrix being responsive to said intensi-
 fying pulses to provide said character dot matrix to
 said cathode ray tube, said synchronization of the 20
 raster scan, vertical scan deflection and intensify-
 ing pulses producing a character representation for
 each stored character which is invariant and at the
 same position on the face of the tube from scan to
 scan. 25
 5. A character generator for providing selected char-
 acters at desired locations on the face of a cathode-ray
 tube comprising:
 means for selecting one of said characters,
 means responsive to said selecting means for provid- 30
 ing a different digital code representation for each
 of a plurality of characters to be generated,
 means for designating the position on the face of the
 cathode-ray tube where said character is to be
 presented, 35
 a rotating magnetic drum containing address posi-
 tions along its periphery corresponding with fixed
 positions on the face of the cathode-ray tube,
 means for storing in the drum address corresponding
 to a position on the face of the tube the digital code 40
 corresponding to the character to be presented at

that position,
 means for reading out the stored digital code of each
 character on each revolution of the drum,
 a cathode ray tube having a scanning beam,
 means for providing a raster scan having a sinusoidal
 vertical deflection to said tube
 a second means for storing binary signals, each corre-
 sponding to a different character code,
 means for providing such second storage means the
 serial binary signal of a character in response to the
 digital code representation of that character pro-
 duced by said rotating drum,
 means for synchronizing the raster scan and sinusoi-
 dal deflection with the serial binary signal so that
 each bit of the serial binary signal derived from
 each stored character code on the drum corre-
 sponds to a fixed position on the face of the tube,
 means for intensity modulating the beam of said cath-
 ode ray tube with said serial binary signal to cause
 said intensity modulated beam to provide a dot ma-
 trix representation of the character stored at the
 corresponding drum address.
 6. The apparatus of claim 5 comprising in addition:
 means for comparing adjacent bits in the serial binary
 signal occurring during the upward sweep of the
 sinusoidal scan to provide an additional intensify-
 ing bit intermediate two intensifying bits of said se-
 rial binary signal.
 7. The apparatus of claim 6 comprising in addition:
 means for storing two such successive serial binary
 signals,
 means for comparing bits of these signals which
 occur in a row of the matrix representation of the
 character to provide an additional intensifying bit
 when intensifying bits occur in the serial binary sig-
 nals corresponding to the row,
 means for providing these additional intensifying bits
 as a serial binary signal during the downward sweep
 of the sinusoidal scan.
 * * * * *

45

50

55

60

65