United States Patent

Bard

[15] **3,675,208**

[45] **July 4, 1972**

[54] EDITING SYSTEM FOR VIDEO DISPLAY TERMINAL

[72] Inventor: **Irving Gary Bard**, Philadelphia, Pa.

[73] Assignee: **Delta Data Systems Corporation**, Cornwells Heights, Pa.

[22] Filed: **May 28, 1970**

[21] Appl. No.: **41,474**

[52] **U.S. Cl.** **340/172.5**, 340/324.1

[51] **Int. Cl.** **G06f 3/14**

[58] **Field of Search** 340/172.5, 324.1

[56] **References Cited**

UNITED STATES PATENTS

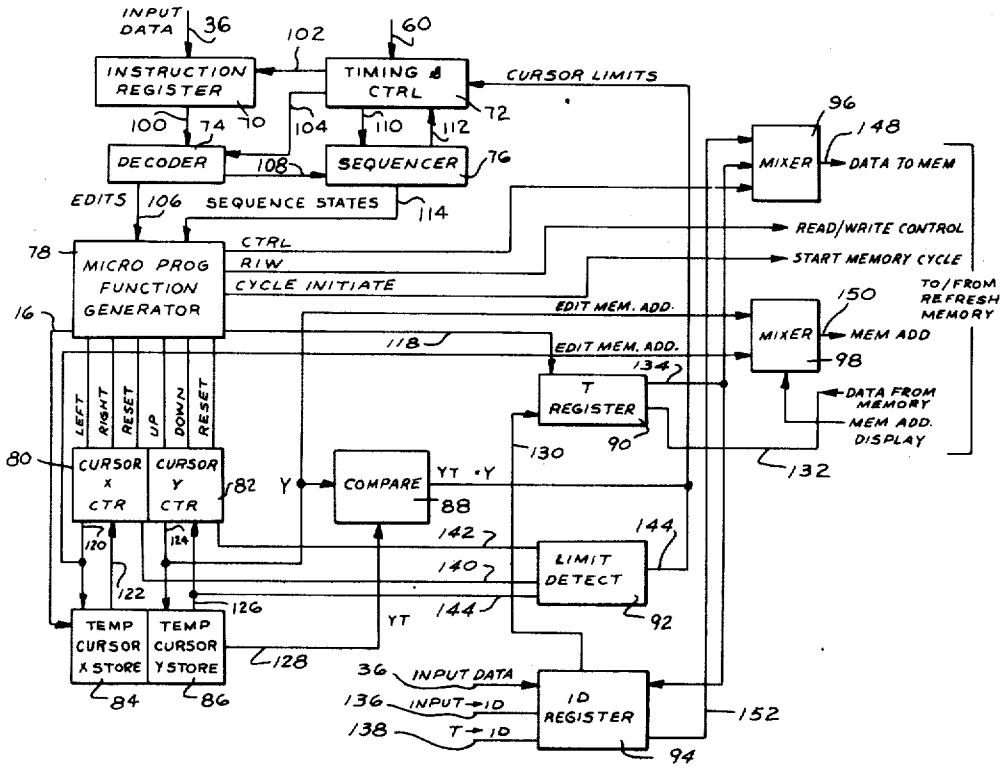
3,555,520	1/1971	Helbig et al.	340/172.5
3,501,746	3/1970	Vosbury	340/172.5
3,364,473	1/1968	Reitz et al.	340/172.5
3,396,377	8/1968	Strout	340/172.5
3,389,404	6/1968	Koster	340/172.5
3,466,645	9/1969	Granberg et al.	340/172.5
3,603,966	9/1971	Gregg et al.	340/172.5
3,191,169	6/1965	Shulman et al.	343/5

Primary Examiner—Paul J. Henon
Assistant Examiner—Mark Edward Nusbaum
Attorney—Caesar, Rivise, Bernstein & Cohen

[57] **ABSTRACT**

A video display terminal is disclosed having an editing system for controlling the entry of characters on the video display as well as the editing or manipulation of the data provided on the display. The editing system is operable between succeeding scan lines of the video display raster. The editing system includes a microprogram function generator which enables the execution of a plurality of steps in an edit by the mere insertion of a single instruction into the video display terminal.

13 Claims, 10 Drawing Figures



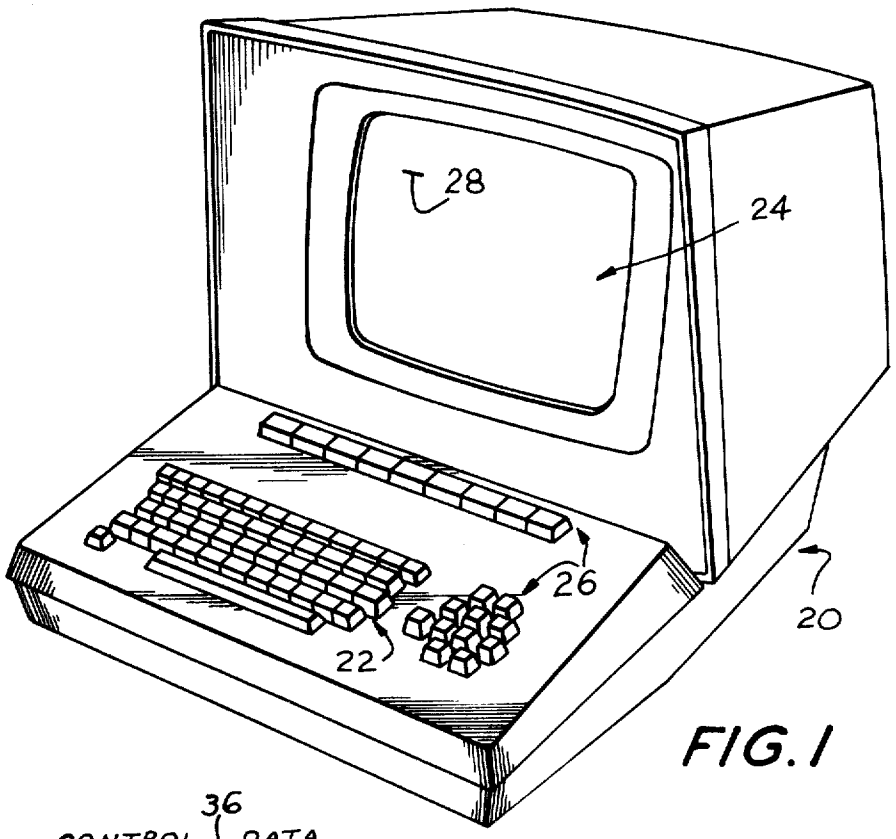


FIG. 1

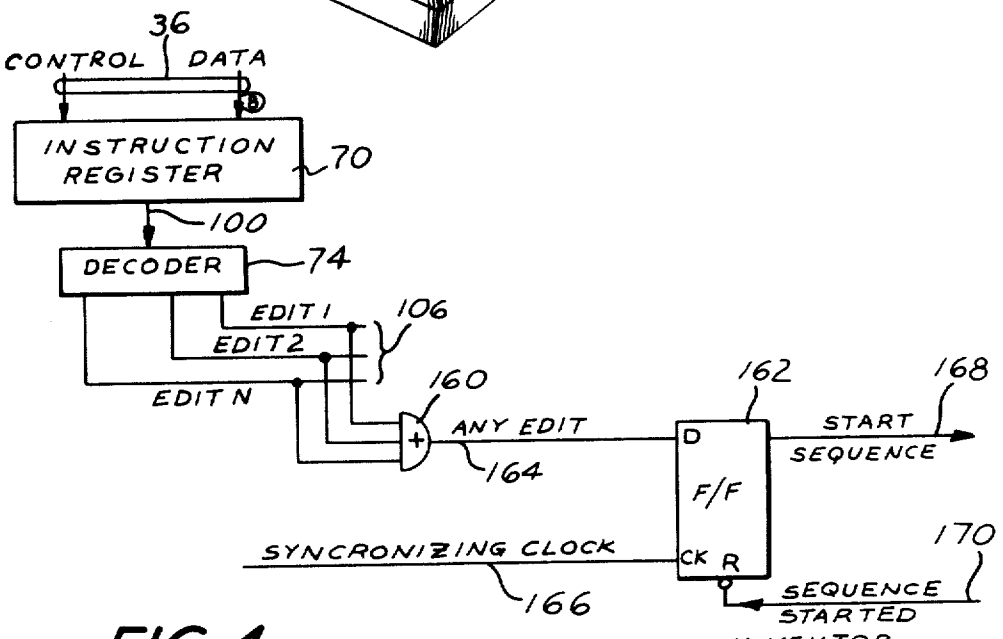


FIG. 4

INVENTOR
IRVING GARY BARD

BY
Caesar, Rivise
Bernstein & Cohen
ATTORNEYS

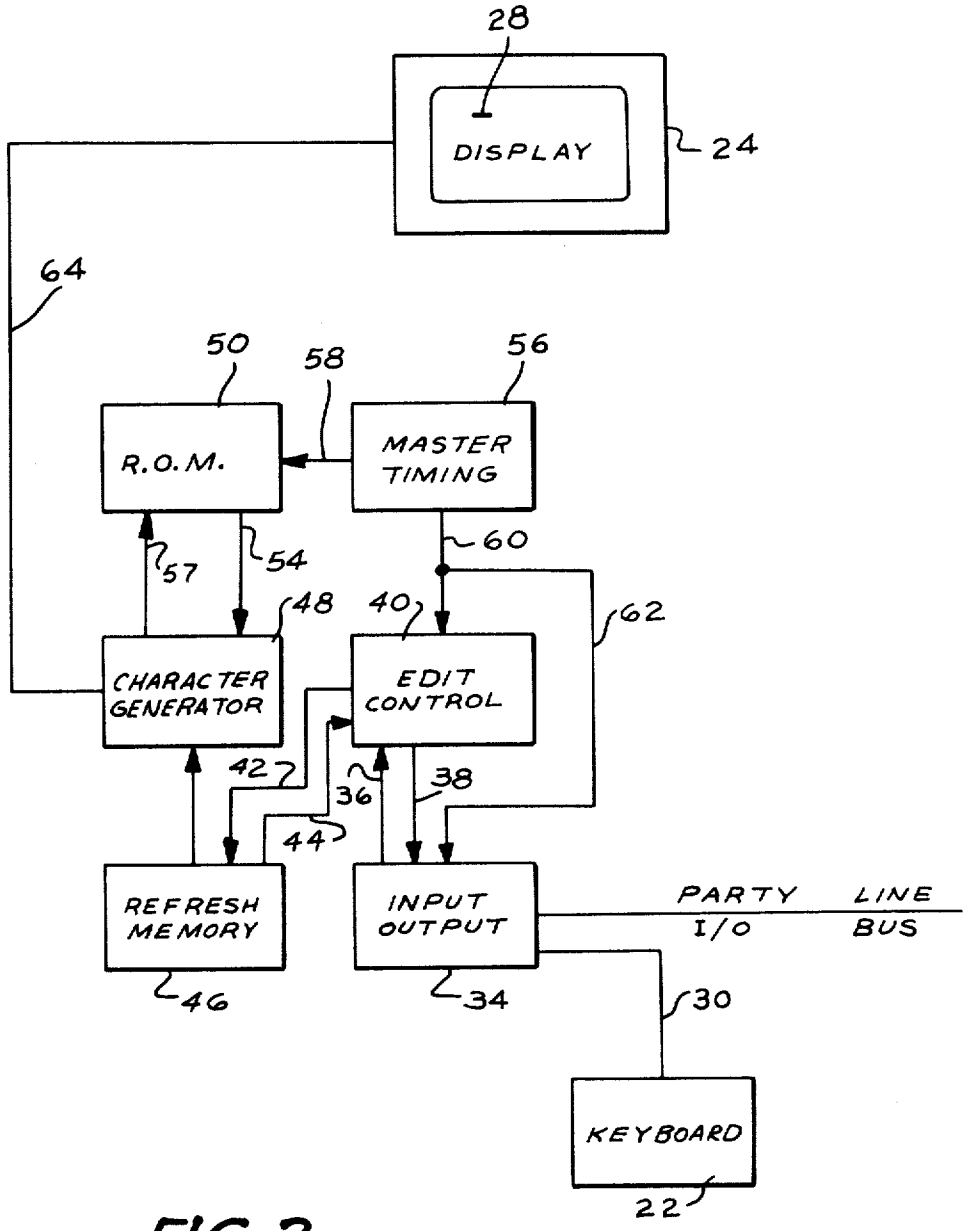


FIG. 2

INVENTOR
IRVING GARY BARD
BY
Caesar, Rivise, Bernstein & Cohen
ATTORNEYS

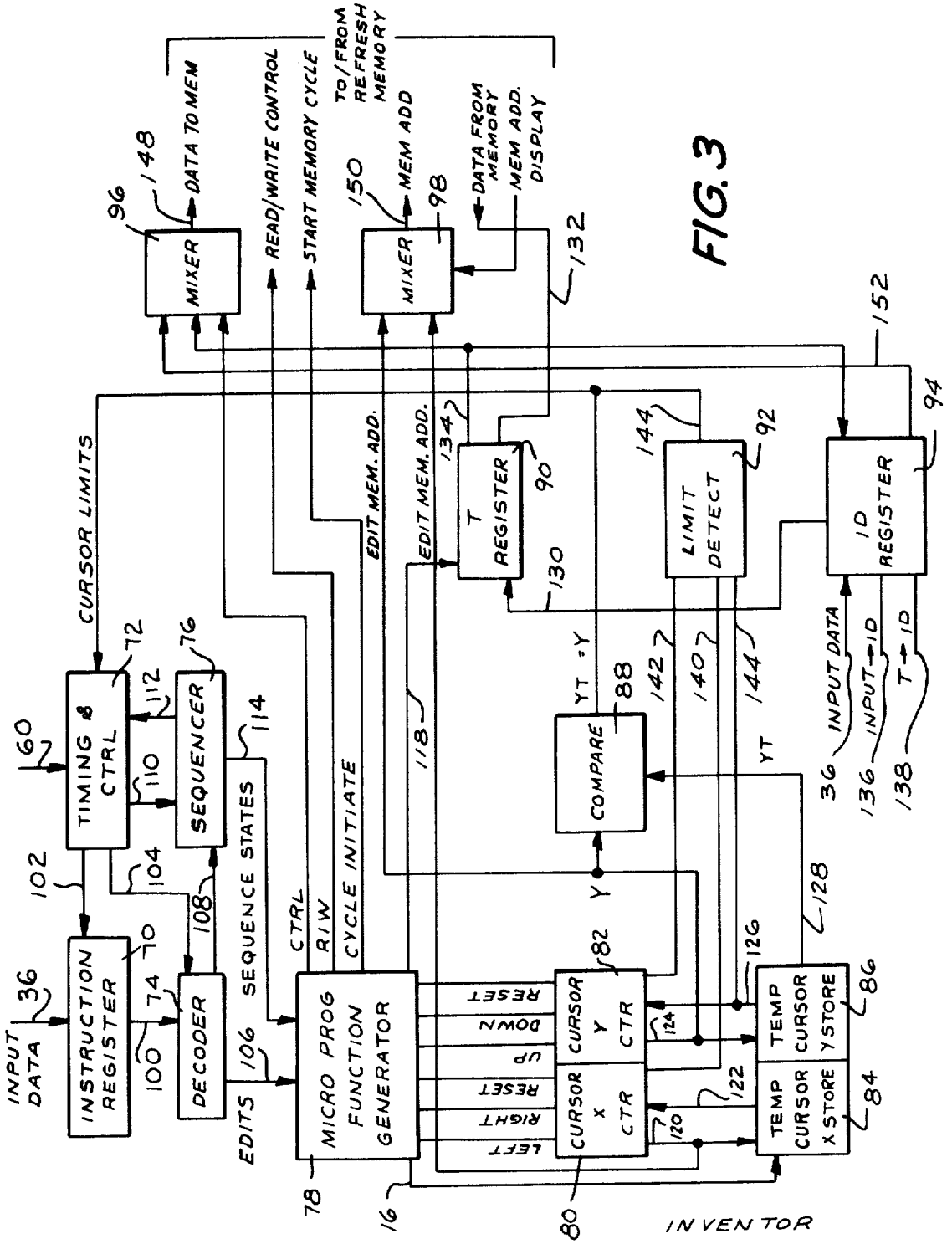


FIG. 3

INVENTOR
IRVING GARY BARD

BY
Caesar, Rivise, Bernstein & Cohen
ATTORNEYS

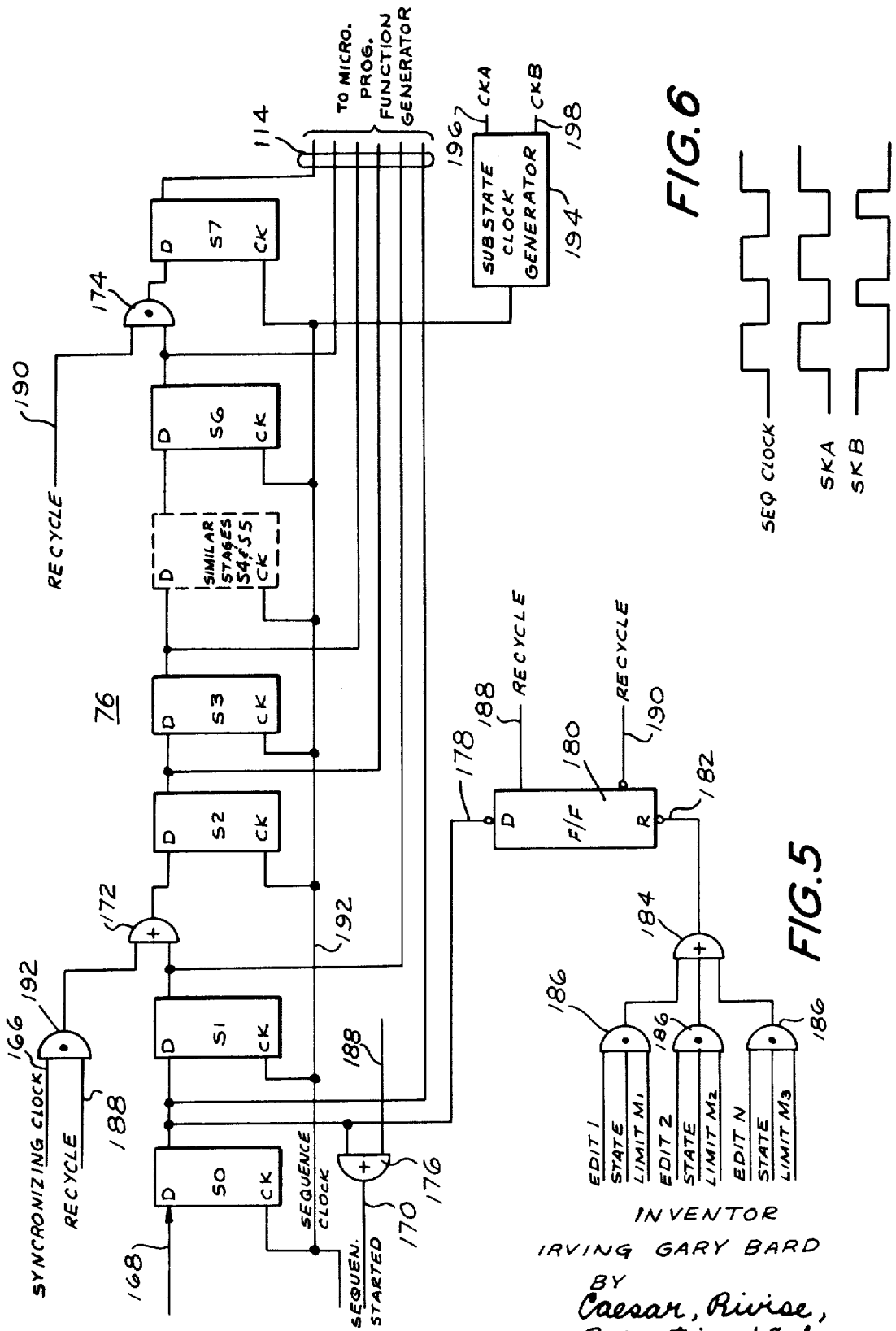


FIG. 5

FIG. 6

INVENTOR
IRVING GARY BARD
BY
Caesar, Rivise, Bernstein & Cohen
ATTORNEYS

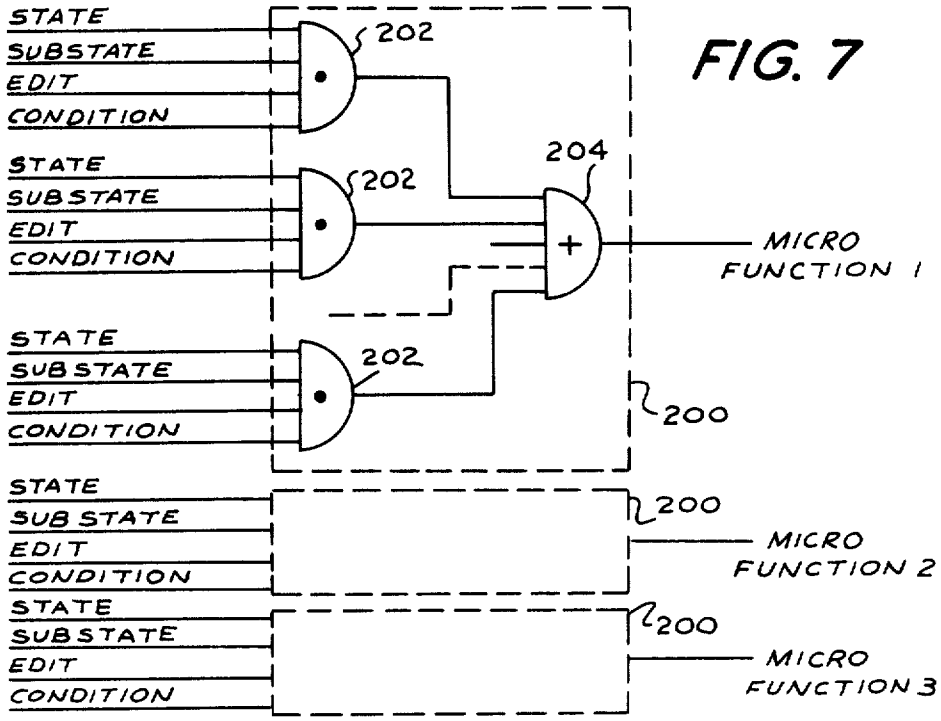


FIG. 7

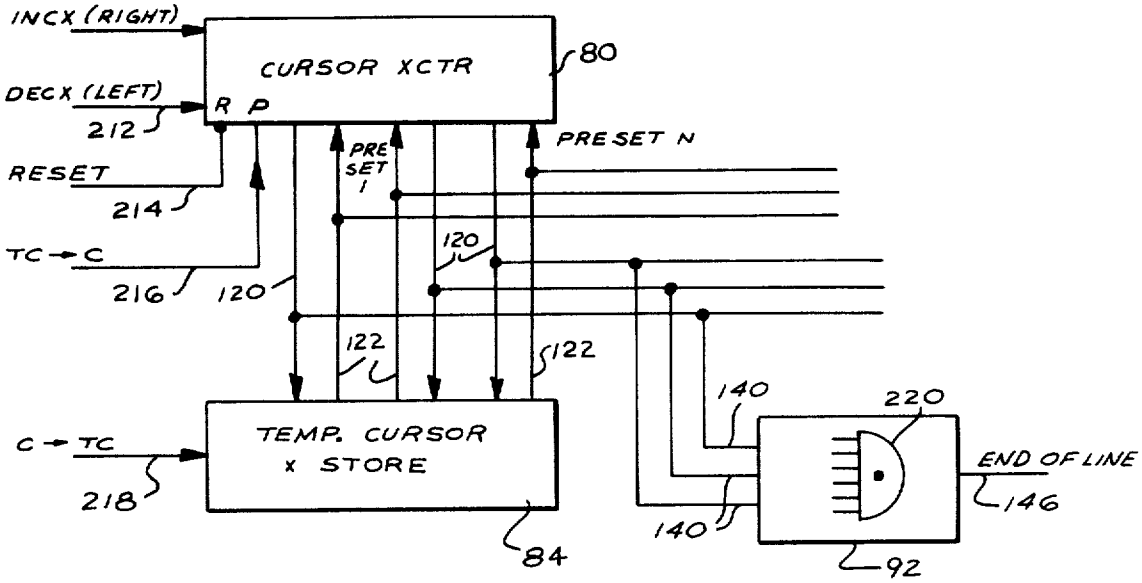


FIG. 8

INVENTOR
IRVING GARY BARD

BY
Caesar, Rivise, Bernstein & Cohen
ATTORNEYS

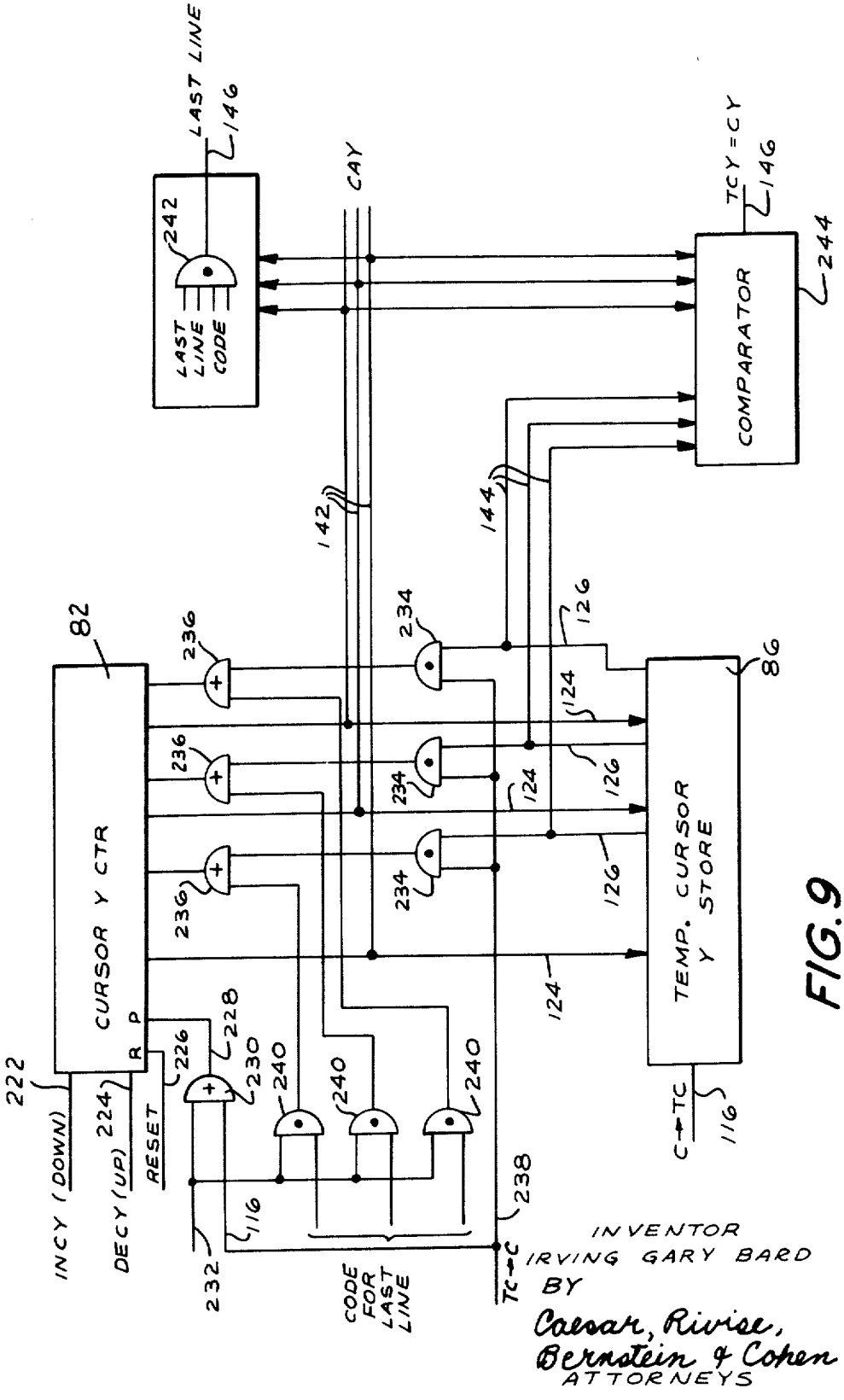


FIG. 9

INVENTOR
IRVING GARY BARD
BY
Caesar, Rivise, Bernstein & Cohen
ATTORNEYS

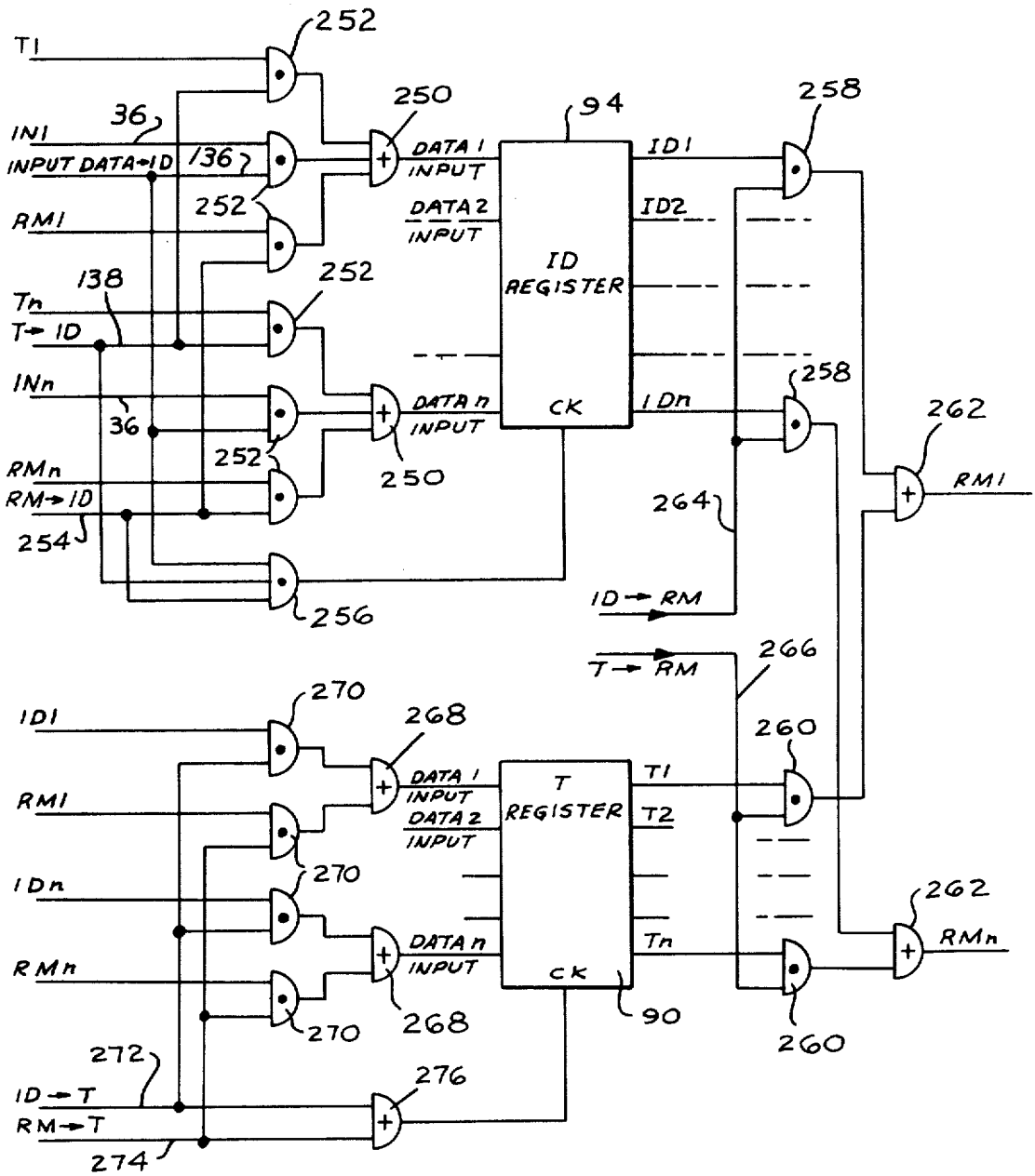


FIG. 10

INVENTOR
IRVING GARY BARD

BY
Caesar, Rivise,
Bernstein & Cohen
ATTORNEYS

EDITING SYSTEM FOR VIDEO DISPLAY TERMINAL

This invention relates generally to video display terminals and more particularly to a video display terminal including an editing system.

The video display terminal has rapidly become a very important peripheral item for central processing units. The video display terminal enables the visual display of data which will be entered into a computer memory or transmitted to a remote terminal prior to the entering or transmittal thereof. This means that verification of the data that is to be inserted into the computer or transmitted is visually available to the operator of the video display terminal.

Moreover, the video display terminal enables the rapid visual inspection of data received from a central processing unit. In addition, with various improvements that have been provided in video display terminals, formats may be provided on the face of the video display terminal which enable the rapid entering of data into a data processing system without the necessity of identifying the nature of the information. It has been found, however, that video display terminals presently available do not have adequate editing capability of the data that is presently displayed on the screen of the video display terminal. That is, if a substantial portion of a page of data has been entered onto the screen of a video display terminal and it is desired to edit the information thereon, it is necessary for the operator to enter each character over the characters presently in the display until the entire display is correct.

In some cases, this is not only time consuming, but it is also impossible to insert a character or delete a character without requiring a complete destruction of the message. For example, if it is assumed that ten lines of data appear on the video display terminal and it is found that a complete line has been omitted between the third and fourth lines on the video display terminal, it would be necessary to completely delete all of the characters on the fourth through tenth lines of the video display terminal. Thus, not only is it necessary to enter the line which should be inserted, but the fourth through tenth line must be re-entered below the inserted line.

Re-entering is not only frustrating for the operator, but it is also time consuming. Similarly, there are cases where it is necessary to insert a character on a line and in order to do so, it is necessary to delete each of the characters on the line starting at the position at which the character is desired and the succeeding characters on the line. Then, not only is the character entered, but all of the characters that have been deleted, re-entered in their proper position.

It is therefore an object of this invention to overcome the aforementioned disadvantages.

Another object of the invention is to provide a new and improved video display terminal which enables immediate access to the refresh memory for the video display terminal for editing the contents thereof.

Another object of the invention is to provide a new and improved video display terminal which enables a single instruction to modify a plurality of character locations on the video display terminal.

Yet another object of the invention is to provide a new and improved editing system for a video display terminal which utilizes microprogramming.

Yet another object of the invention is to provide a new and improved video display terminal which utilizes the time between succeeding ones of the horizontal scan lines of the cathode ray tube to carry out the editing instructions.

These and other objects of the invention are achieved by providing in a video display terminal having input means for receiving data and instructions and a cathode ray tube for displaying data inserted in the terminal by the input means by providing an alpha-numeric representation of the data on a scan raster which is comprised of a plurality of parallel scan lines, and an editing means. The editing means is responsive to the input means and is initiated at the end of the scan line to immediately enter the data and carry out the instructions. The editing means is operable only during the time between the succeeding scan lines on the cathode ray tube.

Other objects and many of the attendant advantages of this invention will be readily appreciated as the same becomes better understood by reference to the following detailed description when considered in connection with the accompanying drawings wherein:

FIG. 1 is a perspective view of a video display terminal embodying the invention;

FIG. 2 is a schematic block diagram of a video display terminal system;

FIG. 3 is a schematic block diagram of the editing control for the video display terminal;

FIG. 4 is a schematic block diagram showing the interconnection between the instruction register and the edit decoder;

FIG. 5 is a schematic block diagram of the sequencer and the cycle limit detector;

FIG. 6 is a diagrammatic representation of the signals generated by the sequence clock and the substate clock generator;

FIG. 7 is a schematic block diagram of the microprogram function generator;

FIG. 8 is a schematic block diagram showing the interconnection between the cursor position counter and the temporary storage register for the cursor position;

FIG. 9 is a schematic block diagram of the cursor line counter and temporary storage register therefor; and

FIG. 10 is a schematic block diagram of the connections to and from the ID register and the T register.

Referring now in greater detail to the various figures of the drawing wherein similar references numerals refer to similar parts, a video display terminal embodying the invention is shown generally at **20** in FIG. **1**.

The video display terminal includes a standard typewriter keyboard **22** for entering data on a cathode ray tube display screen **24**. Also included as part of the keyboard **22** are a plurality of instruction buttons or keys **26** which are utilized to edit the information on the screen **24** as well as provide instructions for the transfer of the data to and from the video display terminal. A cursor **28** is provided on the video display terminal to indicate the position that a character entered will be provided within the field of data illustrated on the screen. The cursor **28** is, of course, movable by a plurality of the instruction keys **26** that are provided on the keyboard so that the position at which data is entered can be controlled.

Referring now to FIG. **2** wherein a schematic block diagram of the video display terminal system is shown. It can be seen that the keyboard **22** is connected via lines **30** to the circuitry of the video display terminal. Lines **30** are directly connected to the input-output means **34** which receives the instructions and the entered characters from the keyboard **22**. A party line Input-Output (I/O) Bus is also connected to the input-output means **34** for connection to external devices. The input-output means **30** is connected via input and output lines **36** and **38** to the edit control **40**. Edit control **40** is connected via input and output lines **42** and **44**, respectively, to a refresh memory **46**.

The refresh memory **46** is connected to a character generator **48** which acts to read out the contents of a read only memory **50**. The character generator is connected to the read only memory by input and output lines **52** and **54**, respectively. A master timing generator **56** is provided for providing via lines **58**, **60** and **62**, timing signals for the control of the flow of data throughout the system.

The output of the character generator **48** is connected via line **64** to the cathode ray tube display **24** which displays in alpha-numeric form the characters provided to the refresh memory via the input-output device **34**.

The overall operation of the video display terminal is as follows:

Data and instructions are provided via keyboard **22** to the input-output means **34**. The input-output device provides both the data and the instructions to the edit control **40** via lines **36**. The edit control provides access via lines **42** to the refresh memory **46** for the insertion of data in the form of character codes to the refresh memory.

The refresh memory is a high speed random access storage unit which is preferably of the magnetic core type. The refresh memory preferably has a capacity to store 1,024 eight bit words. Six of the eight bits of each word are utilized for the code representative of the character which is to be displayed on the cathode ray tube **24**. The other two bits of the word are preferably used for the purpose of storing format information and blink information as well as any information which may be necessary for the use of a color terminal.

The edit control **40**, thus, operates on the refresh memory to perform all of the entering operations of characters into the storage as well as to manipulate the data in the refresh memory in accordance with instructions which are provided by the keyboard **22**. The refresh memory **46** also acts via the character generator **48** to address various ones of the areas of storage in the read only memory **50**.

The read only memory is a permanent storage device which includes a plurality of permanently stored character patterns. The read only memory is programmed to store a pattern of "1"'s and "0"'s which represent the shape of each of the standard 64 characters which can be displayed on the screen. Each character pattern is preferably made up of seven rows of five bits each. The memory is, thus, broken up into 64 times seven words of five bits each. The coded characters in the refresh memory **46** are provided in such a code that the code itself acts to address the character patterns associated therewith for generation via the character generator **48** to the display **24**.

In the preferred embodiment of the invention, the character generator provides from the read only memory **50**, the character patterns to the display **24** on a time division basis. That is, the cathode ray tube display utilizes a scan raster which is comprised of a plurality of horizontal lines across the screen. Each line is caused by the electron beam of the cathode ray tube being moved across the screen. Characters are generated on the screen by blanking and unblanking the cathode ray tube beam as it moves laterally across the screen. The beam moves across the screen from left to right and top to bottom until the bottom line has been completed and then returns to the top of the screen to start another complete scan.

In the preferred embodiment of this invention, the beam moves across the screen (262 ½) times to create the conventional TV raster scan. Successive sweeps skip every other line creating two fields of scan of 262 ½ which are equally spaced up and down the screen. This method of sweeping is known as the interlaced raster scan. In the display of this invention, the information on each field is refreshed identically so that, in effect, only 262 ½ of the lines on the screen are used rather than 525. However, because each field is repeated 60 times per second, the refresh rate for providing the character patterns to the screen assures a flicker-free display. The approximate time for a beam to sweep from one side of the screen to the other to form a horizontal line is 52 microseconds. The time taken between the righthand side of the beam to return to the left-hand side of the screen takes approximately 12 microseconds. The cathode ray tube display **24** utilizes standard synchronization signals for the operation of the scan raster. The synchronization signal is mixed with additional data from the character generator **48** to provide the video signal on the display **24**.

During the 12 microseconds between each of the lines which are provided on the display, the edit control **40** is utilized to enter data from the keyboard to the refresh memory **46** and also to reorganize the information in the refresh memory in accordance with the instructions provided to the edit control.

The video display terminal is adapted to have a caPacity of 960 characters which are grouped into 24 lines of forty characters each. That is, each horizontal line of characters can display forty characters on the display **24** and 24 lines of characters may be displayed simultaneously.

Each line of characters is comprised of seven horizontal scan raster lines. That is, since each character pattern is comprised of a matrix of five by seven blanked and unblanked spots, each horizontal scan raster line includes five points of the character matrix. For example, the first line of a scan raster of a character line includes the upper line of blanked and unblanked portions of the character patterns of each of the characters that appear on the line. Similarly, the second, third, fourth, fifth, sixth, and seventh would, respectively, include the second, third, fourth, fifth, sixth and seventh line of the character pattern for each of the characters on the line. By a time division basis, the character generator **48** provides the blanking and unblanking signals which are utilized on the cathode ray tube beam for providing the character display. The refresh memory **46** provides the address within the read only memory in combination with the timing signal on line **58** in order to provide the necessary portions of the character matrices as the character pattern is composed onto line **64** and provided to the display **24**.

Thus, to insert a character from keyboard **22** to the position in the display **24** indicated by the cursor **28**, the key associated with the character is pressed. The input data is provided to the input-output means **34** which is then provided via the edit control into the refresh memory **46**. The code representative of the character is, thus, stored at the position in the refresh memory corresponding to the position on the display **24** at which the cursor **28** is located. Thus, as soon as the raster of the cathode ray tube in display **24** reaches the position at which the new character has been entered, the character pattern is addressed by the refresh memory and the character generator **48** and read from the read only memory **50** to provide on line **64**, the new character pattern each time the scan raster line passes through the character position.

The edit control **40** automatically increments the control for the cursor **28** so that it is moved to the next position to the right on the character display so that the next character inserted by keyboard **22** is automatically displayed to the right of the previous character inserted.

The edit control system is schematically illustrated in FIG. **3**. As best seen in FIG. **3**, the edit control includes an instruction register **70**, timing and control unit **72**, a decoder **74**, a sequencer **76**, a microprogram function generator **78**, a cursor X counter **80** and a cursor Y counter **82**, a temporary cursor X store register **84** and a temporary cursor Y store register **86**, a comparator **88**, a T register **90**, a limit detector **92**, an ID register **94** and mixers **96** and **98**.

The instruction register **70** is connected to decoder **74** via lines **100**. The instruction register **70** also receives signals via lines **102** from the timing and control unit **72**. The timing and control receives signals from the master timing unit via line **60** and provides the control signals for enabling the operation of the edit control during the period between the horizontal lines of the display **24**.

The decoder **74** also receives signals via lines **104** from the timing and control unit **72**. The decoder **74** decodes the instruction and character code to provide an output signal on one of output lines **106**. The signal on one of lines **106** provides the microprogram function generator **78** with the necessary signal to initiate the instruction from the instruction register **70**.

The decoder **74** is connected via line **108** to sequencer **76**. Sequencer **76** is also connected via input and output lines **110** and **112**, respectively, to the timing and control unit **72**. The sequencer **76** basically provides a sequence of signals which effectively divide the period between horizontal scan lines into sub-intervals or sequence states for carrying out predetermined ones of said microprogram functions which comprise an instruction from the decoder **74**.

The sequence states are provided on sequentially energized ones of lines **114** which are connected from the sequencer to the microprogram function generator **78**. The microprogram function generator includes a plurality of output lines which are respectively labeled in accordance with their function. For example, the legend "CTRL" stands for control and the line associated therewith is connected to the mixer **96** to enable

the control of the flow of "data to the memory" which refers to the refresh memory. The "R/W" legend indicates the signal on the line associated therewith effectuates a read or write control function to the refresh memory. The "cycle initiate" legend indicates that the signal provided on the line associated therewith to the refresh memory starts the memory cycle.

Similarly, the output lines from the microprogram function generator labeled "left," "right" and "reset" which are connected to the cursor X counter effectively indicate that the counter is decremented, incremented and reset, respectively, by the signals on the associated lines therewith.

The lines connected to the cursor Y counter **82** and having the legends "up," "down" and "reset" associated therewith are, respectively, decrement, increment and reset lines of the cursor Y counter. The microprogram function generator also includes an output line **116** which is connected to the temporary cursor X store and the temporary cursor Y store and acts to transfer the contents of these stores to the cursor X counter and cursor Y counter, respectively, when a signal is provided on line **116**.

Similarly, a line **118** connected from the microprogram function generator **78** to the T register **90** causes the transfer of data into the T register **90**. Additional output lines are also provided from the microprogram function generator to the various components throughout the system which are seen in greater detail hereinafter with respect to the more detailed drawings of the connections between various ones of the components of the system.

The cursor X counter is connected to the temporary cursor X store **84** via input and output lines **120** and **122**. Similarly, the cursor Y counter is connected to the temporary cursor Y store **86** via lines **124** and **126**. Lines **120** and **124** which are the output lines from the cursor X counter and the cursor Y counter, respectively, are also connected to the mixer **98** and, as indicated by the legend "Edit Mem. Add." thereon act to access the address in the refresh memory at which the contents are altered.

Output lines **124** from cursor Y counter **82** are also connected to the comparator **88**. The temporary cursor Y store register **86** is also connected to comparator **88** via output lines **128**.

The T register **90** includes input lines **130** from ID register **94** and lines **132** from the refresh memory. The output lines **134** of the T register **90** are connected to the mixer **96** which is the input means for and provides the information to the refresh memory and to the input of the ID register **94**.

The ID register includes, in addition to the input lines from the T register **90**, input lines **36** from the input-output means **34**, line **136** which is provided by the microprogram function generator **78** which, when energized, acts to transfer the data on lines **36** into the ID register **94** and line **138** which is also connected to the microprogram function generator and which provides a signal thereon when the contents of the T register **90** are sent to the ID register. It should be noted that the legends "input → ID" and "T → ID" represent, respectively, a transfer of data from the input source to the ID register and from the T register to the ID register. Legends are provided similar to these throughout the drawings and indicate a flow of data from one register to the other when a signal is provided on that line to the register.

The limit detector **92** includes input lines **140** and **142** which are connected, respectively, to the cursor X counter and cursor Y counter **80** and **82**, respectively. The limit detector **92** includes means for comparing the counts in the counters **80** and **82** with predetermined limits which are determined by the number of characters which can be stored on a line and the number of lines of characters which can be provided on the display. Thus, the limit detector **92** is provided with output lines **144** which are utilized to provide to the timing and control units **72** signals thereon indicative of whether the cursor X counter or the cursor Y counter have a count therein indicative of either the last character position on a line or the last line in the character display. The limit detector **92** also provides signals on the output lines **144** to the timing and control **72** which are indicative of other limits such as the temporary cursory store having the same count thereon as that in the cursor Y counter **82**.

The mixers **96** and **98** act to provide the buffer storage necessary for insertion of characters into the refresh memory. Thus, mixer **96** is connected via output line **148** to the refresh memory and mixer **98** is connected via output line **150** to the refresh memory. As set forth above, the mixer **96** receives the contents of the T register **90** from lines **134** when the microprogram function generator provides a signal on line **118** which transfers the contents of the T register to the refresh memory. The mixer **96** also receives input signals on lines **152** from the ID register **94**.

Referring to FIG. **4**, it can be seen that the input lines **36** to the instruction register **70** are comprised of control lines and eight data lines. The control lines are actuated not only when an instruction is given to the video display terminal, but also when data signals are provided, the control lines have an instruction thereon indicative of the fact that a character is to be entered.

The decoder **74** receives the signals from lines **100** and converts the same into a "1 out of N" code by providing a signal on one of the edit lines edit **1** through edit N. Connected to the lines **106** which include the editing signals is an OR gate **160** which is responsive to all of the edit lines so that if a signal is provided on any of the edit lines, the OR gate is enabled and provides a signal to flip-flop **162**.

The OR gate **160** is connected to the set input of flip-flop **162** via line **164**. Thus, as indicated by the legend "any edit" appearing above line **164**, any editing signal provides a setting signal for the flop-flop **162**. It should be noted that the flip-flop **162** includes the notation "D" adjacent the input line **164**. The legend D in each of the flip-flops shown throughout the circuitry indicates the set input. The legend "CK" indicates that a line connected thereto acts to provide a triggering function for the flip-flop each time a signal is provided on the line.

Line **166** is connected to the clock or trigger input CK of flip-flop **162** to effectively cause a change of state in the flip-flop in accordance with the signal provided on line **164**. That is, as soon as an instruction is received via lines **36** to the instruction register **70**, the decoder **74** provides a continuous signal on one of the lines **106** from the decoder. As long as one of the lines **106** remains high, OR gate **160** remains enabled thereby causing a positive signal to be provided to the set input of the flip-flop **162**.

The flip-flop **162** is not set, however, until such time as the leading edge of a pulse on line **166** triggers the flip-flop **162** to change its state. Thus, if the signal on line **164** is high, it causes the flip-flop to change in state to a 1. If the flip-flop is already in the 1 state, the flip-flop remains unchanged on the next triggering input signal on line **166**. The signals on line **166** are provided by the synchronizing clock in the timing and control unit **72** which is responsive to the pulses at the end of each horizontal scan raster line in the video display.

As soon as flip-flop **162** is set, output line **168** thereof goes high thereby enabling the sequencer **76** to be started. The flip-flop **162** also includes an input line **170** which is connected to the reset input thereof and causes the flip-flop to be changed to the 0 state when a flip-flop is provided thereon.

Referring now to FIG. **5**, it can be seen that the sequencer **76** is comprised of a shift register having eight flip-flop stages which are labeled, respectively, SO through S7. Each of the flip-flops S0 through S7 includes a set input (D) and a clock input (CK). Stages S4 and S5 of the shift register are indicated in phantom since both stages are similar and are connected in tandem between S3 and S6 in sequence. The output lines of each of the shift register stages S0 through S7 are connected to one of the output lines **114** which are connected to the microprogram function generator.

The output line of each of the shift registers is connected to the set input of the next succeeding stage of the shift register.

That is, the output of shift register stage SO is connected to the set input of flip-flop S1. The output of flip-flop S1 is connected to the set input of S2 via an OR gate 172. The output of shift register S2 is connected to the set input of shift register stage S3 and so on through S6 which is connected to the set input of shift register S7 via AND gate 174. Stages S3 through S6 are connected together as is stage S2 to S3.

The output line of flip-flop S0 is also connected to a first input of OR gate 176. The output of OR gate 176 is connected to line 170 which is connected to the reset input of flip-flop 162 in FIG. 4. The signal on output line 170 indicates that a sequence has been started in the shift register which forms the sequencer 76. The output line of flip-flop S0 is also connected via line 178 to the set input line of flip-flop 180. The flip-flop 180 also includes a reset input line 182 which is connected to the output of an OR gate 184. OR gate 184 includes a plurality of inputs from the output of a plurality of AND gates 186. Each of the AND gates 186 is connected at one of its input lines to one of the edit 1 through edit N lines. Thus, the top AND gate 186 in FIG. 5 includes a line which is connected to edit 1, the next AND gate 186 includes an input line from edit 2 and so on through the lowermost AND gate 186 which includes an input from the edit N line.

Each of the instructions has limits incorporated therein which end the sequencing function of the shift register 76 to initiate various ones of the microprograms. That is, each of the main instructions normally utilizes a plurality of complete cycles of the sequence shift register 76. However, when a predetermined condition has been reached, the limits are detected and provide signals to AND gates 186 to indicate that a particular editing instruction has been substantially completed. Thus, a high signal will be provided on each of the lines to the AND gates 186 associated with the particular instruction. The line associated with the legend "state" indicates the portion of the sequence in which these limits are detected. That is, each of the microprogram functions are carried out in predetermined portions of a cycle.

Thus, if a test is made of the status of the circuitry to determine the end of the instruction associated with the edit 1 signal during the S2 state of the sequencing cycle, the line of lines 114 connected to the output of flip-flop S2 is connected to the topmost AND gate 186 in FIG. 5.

Thus, if any of the AND gates 186 is enabled by a high signal on each of its input lines, the OR gate 184 receives an enabling signal which causes the output line to reset the recycle flip-flop 180. When the flip-flop 180 is in the set state, a high signal is provided on output line 188 which is labeled the recycle signal. When the flip-flop 180 is reset, the signal on output line 190 is made high thereby indicating that there should not be a recycle as evidenced by the legend "$\overline{\text{recycle}}$" having a bar, or the logical not sign, over the top thereof.

The OR gate 172 also includes an input from the output of AND gate 192. The AND gate 192 includes a first input from line 188 of flip-flop 180 and a second input line 166 which receives the signal from the synchronizing clock which provides the signal at the end of each of the horizontal scan lines on the cathode ray tube display. Line 188 is also connected to the second input of OR gate 176 and prevents the sequence flip-flop 162 (FIG. 4) from being set again during an instruction edit.

The output line 190 of flip-flop 180 which is labeled recycle is connected to the second input of AND gate 174 which is connected between stages S6 and S7 of the shift register 76 of the sequencer. To each of the clock inputs of the flip-flop stages S0 and S7 of the shift register 76 is connected input line 192.

Line 192 receives the signals from the output of the sequence clock which is provided in the timing and control unit 72. The sequencing clock line 192 is also connected to a substate clock generator 194 which provides, on the output of lines 196 and 198, signals CKA and CKB, respectively. These output signals are illustrated in FIG. 6. It can be seen that the sequence clock signal is a square wave pulse. In response to the sequence clock signal, the substate clock generator generates clock signals CKA and CKB which act effectively to divide the sequence state into two substates and B, respectively.

The sequence clock enables the set state of flip-flop 162 in FIG. 4 to cause a 1 to be shifted through the stages S0 through S7 of the shift register 76. That is, when the flip-flop 162 is set, a 1 signal is provided on line 168 to the set input of shift register stage S0. The first clock pulse from the sequence clock on line 192 causes the shift register stage S0 to be changed to the 1 state in accordance with the 1 signal provided on line 168. As soon as the shift register stage S0 becomes a 1, the output line thereof goes high thereby causing the OR gate 176 to be enabled and causing line 170 to reset the flip-flop 162. Thus, the output line of the flip-flop 162 goes low thereby causing the input line 168 to flip-flop stage S0 to go low. Accordingly, on the next clock pulse, flip-flop S0 is switched to the 0 state but because the output line of S0 had been high on the same clock pulse, flip-flop S1 is changed from the 0 to the 1 state. On the next clock pulse, the 1 in flip-flop S1 is shifted to the flip-flop S2 via OR age 172. Upon the next succeeding clock pulses, the 1 is shifted from the stages S2 through S6. If AND gate 174 is enabled by a high signal on line 190, the next pulse causes the 1 to be shifted into S7. Otherwise, the 1 is lost. It should be noted that after the 1 has been shifted out, a 0 has been placed in each of the flip-flop stages due to the fact that the passed stages are in the 0 state.

As soon as the flip-flop S0 was changed in state to the 0 state, the output line thereof caused the flip-flop 180 to be set to cause a high signal on output line 188 which indicated that a recycle of the shift register should take place unless during the shifting of the 1 through to shift register stage S6, the flip-flop 180 was not reset by the occurrence of a condition during one of the states S1 through S6. If the flip-flop 180 is not reset prior to the shifting of the 1 into flip-flop S6, the 1 is lost during the next sequence clock pulse because of the fact that the AND gate 174 cannot be enabled since the $\overline{\text{recycle}}$ input line 190 is at a low potential and the AND gate 174 is not enabled by the setting of flip-flop 7. The 1 is thus shifted out of the shift register 76. The editing system is then inactive until the next synchronizing clock pulse is provided at the end of the next horizontal scan raster line. As soon as the next horizontal scan raster line is completed, the synchronizing clock signal is provided on line 166 to AND gate 192 which thereby causes gate 192 to be enabled since the recycle line 188 is high. A 1 is then inserted in flip-flop S2 upon the next sequence clock pulse. The 1 is then shifted from shift register stage S2 through S6 with the flip-flop 180 remaining in the set state unless a limit has been reached indicating that there should not be a recycle. When the limit has been reached during states S2 through S6, the flip-flop 180 is reset thereby causing gate 174 to be enabled with the flip-flop S6 is set. This causes the 1 to be shifted from the flip-flop S6 to flip-flop S7 which causes the output line of the flip-flop S7 to initiate the termination of an instruction.

The microprogram function generator is shown schematically in FIG. 7. The microprogram function generator basically comprises a plurality of gates 200, each of which is logically connected to produce an output signal when a specific microfunction is to be carried out. Each of the gates 200 includes a plurality of AND gates 202, the output of which are each connected to the input of an OR gate 204.

Where no predetermined condition is required for a microfunction to be carried out, the AND gate 202 would not have a condition input. For example, if microfunction 1 enables the setting of the temporary cursor storage registers 84 and 86 to the present contents of the counters 80 and 82, the condition input to the AND gates 202 is not necessary. Accordingly, each instruction that uses this microfunction 1 has an AND gate 202 provided in gate 200 therefor. In addition, the state and substate lines connected to the inputs of AND gates 202 are connected to the outputs of the sequence shift registers 76 in accordance with the state and substate of the

sequence at which the microfunction should be carried out. Accordingly, if in the instruction associated with edit 1, the microfunction 1 is to be carried out during the time that the flip-flop S1 is in the 1 state, and during substate A thereof, the state line to AND gate 202 is connected to the output of flip-flop S1, the substate line is connected to the CKA line and the edit line is connected to the edit 1 line of lines 106. Thus, if each of the lines is high, the AND gate 202 causes the microfunction 1 output line to be high. Thus, it can be seen that the AND gates 202 are provided for each of the conditions that require a specific microfunction to be carried out. An OR gate is provided at the output of the AND gates so that any one of these conditions satisfies the requisite for carrying out the function. Similarly, in each of the other gates 200 for the other microfunctions that are to be carried out, AND gates 202 are provided in accordance with the number of conditions which require the execution of the microfunction.

In FIG. 8, the connection between the cursor X counter and the temporary cursor store is shown. The cursor X counter 80 includes a first input line (INCX) from the microprogram function generator which increments the counter and thereby effectively changes the position of the cursor by moving it to the right. similarly, the line 212 (DECX) from the microprogram function generator enables the cursor X counter to be decremented which effectively means that the position of the cursor is moved to the left on the display. Line 214 (RESET) from the microprogram function generator 78 enables a signal thereon to reset the counter which effectively puts the cursor at the start of the line.

Line 216 (TC → C) is connected to the preset input of cursor X counter 80 and effectively causes the contents of the temporary cursor storage register 84 to be placed into the cursor X counter 80. Line 218 (C → TC) from the microfunction generator to the temporary cursor X store 84 effectively causes the transfer of the contents of the cursor X counter to the temporary cursor X store.

Each of the stages of the cursor X counter 80 and the temporary cursor X storage register 84 are connected to each other via lines 120 and 122 so that the contents in either of the registers can be preset in the other register.

Lines 140 which extend to the limit detector 92 are connected to the output lines 120 of the cursor X counter 80. The limit detector 92 includes coincidence gates for each of the limits that it is to detect. For example, to determine the end of a line, the gating is so connected to lines 140 that only the code representative of the last character in a line can enable the gate 220 provided in the limit detector which indicates the end of a line. For example, if 40 characters can be stored on a line, the coded representation of the number 40 is detected (where the positions are numbered one through 40) by the AND gate 220 which provides an enabling signal on the output line 144 to indicate that the cursor is at the end of a line as indicated by the contents of the cursor counter.

The connection between the cursor Y counter and the temporary cursor Y storage 86 is illustrated in FIG. 9. The input lines to the cursor Y counter 82 include line 222 (INCY) from the microprogram function generator which enables an incrementing by one of the cursor Y counter each time a pulse is received on line 222. This effectively causes the position of the cursor to be moved down on the display.

Similarly, line 224 (DECY) from the microprogram function generator enables a signal thereon to decrement the Y counter which effectively causes the cursor to be moved upwardly on the display. A signal on input line 226 (RESET) to the cursor Y counter causes the counter to be reset which effectively causes the cursor to be moved to the top line.

Counter 82 also includes an input line 228 which is connected to the output of OR GATE 230. The OR gate 230 has two inputs, one of which is connected to line 116 from the microprogram function generator which enables the contents of the temporary cursor Y store 86 to be preset into the cursor Y counter 82. The second input line 232 to the OR gate 230 is connected to the microprogram function generator 78 to enable the transfer of the code for the last line of characters to be inserted into the cursor Y counter 82.

Each of the output lines of the stages of the cursor Y counter 82 are connected via lines 124 to the input of the stages of the temporary cursor Y store 86. The output lines of the temporary cursor Y store are connected via lines 126 to a plurality of AND gates 234. Connected to the input of each of the stages of the cursor Y counter 82 are OR gates 236. Each of the OR gates 236 has one input connected to the output of AND gate 234 from the stage associated therewith of the temporary cursor Y store 86.

Output line 238 (TC → C) from the microfunction generator is utilized to enable the contents of the temporary cursor Y store 86 to be preset in the cursor Y counter 82. Line 238 is connected to the second input of each of the AND gates 234. Thus, each time line 238 is enabled by the microprogram function generator 78, AND gates 234 are enabled in accordance with the contents of the temporary cursor Y store to preset the cursor Y counter 82.

The second input line of each of the OR gates 236 is connected to the output of a plurality of AND gates 240. The AND gates 240 are connected to positive voltage and ground in accordance with the code for the last line of the display. That is, since the display has a capacity of 24 lines, the binary code for 24 is connected via AND gates 240 to OR gates 236 so that a pulse on line 232 causes the cursor Y counter to be preset to the count for the last line thereby setting the cursor into the last line on the display.

It should be understood that although only the gates and inputs for three stages of the cursor X and Y counters are shown herein for purposes of clarity, each of the counters 80 and 82 and the storage registers 84 and 86 preferably include six stages. The remaining stages are, of course, connected in the same manner as are the three stages shown in FIGS 8 and 9.

The output lines 124 of the cursor Y counter 82 are connected via lines 142 to the limit detector 92 which includes gating 242 which is adapted to be enabled if the counter has a count equal to the number of the last line of the display. Thus, in the preferred embodiment, when the code on lines 142 is equal to the decimal number twenty-four, an output signal is provided on one of the lines 146 indicative of the fact that the last line is in the cursor Y counter.

Lines 142 are also connected to a comparator 244 which is provided in the limit detector. Also connected to the comparator 244 via lines 144 are the output lines 126 of the temporary cursor Y store 86. When the code on lines 144 is similar to the code on lines 142, the comparator 244 indicates that this condition is so by providing an enabling signal on one of lines 146 to indicate that the contents of the temporary cursor Y storage is equal to the contents of the cursor Y counter.

The transfer of data into and out of the T register and the ID register is shown schematically in FIG. 10. Both the ID register 94 and the T register 90 preferably comprise eight flip-flop stages for storage of the coded word. The inputs to each of the stages are, respectively, entitled "data 1 input" . . . "data *n* input." Similarly, the T register 90 has each of its inputs labeled "data 1 input" through . . . "data *n* input." The input labeled "CK" in both the ID and T registers indicates that it enables the stages of the register to be changed in accordance with the input provided to the set input of each of the flip-flops.

A plurality of the OR gates 250, one for each of the data inputs to ID register 94, are provided. Each of the OR gates is connected to one of the inputs of the ID register. Each of the OR gates 250 is connected to the outputs of AND gates 252. One of each group of three AND gates is connected to the output of one stage of the T register 90. One of the AND gates of each of the groups of three AND gates is connected to one of the input lines 36 of the data input and the third one of each of the group of three AND gates is connected to one of the output lines from the refresh memory.

The input lines from the output of the T register 90 are, respectively, labeled T1 . . . Tn, the input data lines are labeled IN1 . . . INn, and the refresh memory lines are labeled RM1 . . . RMn, respectively.

Each of the AND gates **252** is a two input AND gate. Line **138** (T → ID) is connected to the other input of the AND gates which are connected to the output of the T register. Line **138** is connected to the microprogram function generator and enables the transfer of the contents of the T register to the ID register. Line **136** (INPUT DATA → ID) from the microfunction generator is connected to each of the AND gates which are connected to the lines **36**. Line **136** enables the microprogram function generator to transfer the input data to the ID register **94**.

Line **254** (RM → ID) is connected to the other input of each of the AND gates which are connected to the output of the refresh memory. Line **254** is also connected to the microprogram function generator and enables the microprogram function generator to transfer the contents of the addressed location of the refresh memory to the ID register.

An OR gate **256** is also provided, the inputs of which are connected to lines **136**, **138** and **254**. The output of OR gate **256** is connected to the trigger input (CK) of the ID register so that a pulse on either of lines **136**, **138** or **254** enables ID register **94** to store the inputs provided on the data **1** through data *n* input line.

The output lines of each of the stages of ID register **94** are, respectively, labeled ID1 through IDn. Each of these lines are connected to one input of one of a plurality of AND gates **258**. The output lines from each of the stages of the T register **90** are, respectively, labeled T1 through Tn. Each of these output lines are connected to one of a plurality of AND gates **260**. A plurality of OR gates **262** are provided, each of which is connected to the output of one AND gate from the group **258** and one AND gate from group **260** which is associated with one stage of each of the registers. That is, the AND gates **258** and **260** which are associated with the first stage of the ID register and the T register are connected to a first one of the OR gates **262** and so on through to the AND gate **258** which is associated with the *n* stage **94** and AND gate **260** which is associated with the *n* stage of T register **90** which are connected to the *n*th OR gate **262**.

The output of the OR gates **262** are connected to the first through *n*th bits of the input register in the refresh memory. The contents of the input register of the refresh memory are then stored at the location specified in the cursor X and cursor Y counters **80** and **82**. The ID register contents are transferred to the refresh memory when a pulse is provided on line **264** (ID → RM) which is connected to the other input of each of the AND gates **258**. Line **264** is connected to the output of the microprogram function generator **78** and acts to transfer the contents of the ID register to the refresh memory when an enabling signal is provided thereon. Similarly, line **266** (T → RM) is connected to each of the second inputs of the AND gates **260** and is also connected to the microprogram function generator which provides an enabling signal to transfer the contents of the T register to the refresh memory.

Connected to each of the data **1** through data *n* inputs of the T register **90** are a plurality of OR gates **268**. One OR gate is provided for each of the inputs to the T register. Each OR gate is connected to the outputs of a plurality of AND gates **270**. One input of one of the AND gates of each of the pairs of AND gates **270** is connected to one of the output lines of the ID register **94**. One input of the other of the pairs of AND gates is connected to the output lines of the refresh memory. Line **272** (ID → T) which is connected from the microprogram function generator is connected to each of the AND gates associated with the output line from the ID register **94**. A signal on line **272** enables the transfer of the contents of the ID register to the T register. Line **274** (RM → T) is connected to each of the AND gates which are connected to the outputs of the refresh memory. Line **274** is also connected from the microprogram function generator and an enabling signal thereon causes the transfer of the refresh memory location at the address of the cursor to be transferred to the T register **90**. Lines **272** and **274** are connected to the inputs of an OR gate **276**, the output of which is connected to the trigger input of the T register **90**. Thus, when an enabling signal is provided on either line **272** or **274**, the T register **90** is triggered to be set to the input provided on the output lines of the OR gates **268**.

As seen hereinabove, the combination of the editing instruction provided by the instruction register to the decoder and in turn to the microprogram function generator and the sequence states provided by the sequencer **76** to microprogram function generator cause various microfunctions to carry out a complete instruction. The following is a microfunction table broken down into the areas of control, namely, the memory control, the cursor control and miscellaneous control with an abbreviation of the function to the right thereof:

MICROFUNCTION TABLE

MEMORY CONTROL	ABBREVIATIONS
Start memory cycle	CYIN
Select either a read cycle or a write cycle	READ or WRITE
Transfer data in memory to the T register	RM→T
Transfer data in memory to ID register	RM→ID
Preset the memory input to the contents of the ID register	ID→RM
Preset the memory input to the contents of the T register	T→RM
CURSOR CONTROL	
Reset cursor X counter (start of line)	RCX
Reset cursor Y counter (top line)	RCY
Preset cursor Y counter (bottom line)	PCY
Increment cursor X counter (move to right)	INCX
Decrement cursor X counter (move to left)	DECX
Increment cursor Y counter (move cursor down)	INCY
Decrement cursor Y counter (move cursor up)	DECY
Preset the cursor X counter and cursor Y counter with the contents of the temporary cursor X store and Y store registers, respectively	TC→C
Preset the temporary cursor X storage register and the Y storage register to the preset count in the cursor counter and cursor Y counter	C→TC
MISCELLANEOUS CONTROL	
Transfer the contents of the T register to the ID register	T→ID
Transfer the contents of the ID register to the T register	ID→T
Insert code for blank space in T register	BLANK→T
Insert code for blank space in ID register	BLANK→ID
Reset instruction register including stage associated with the instruction	END EDIT

Each of the instructions which can be initiated by pressing one of the keys on keyboard **22** is comprised of a plurality of the above microfunctions and each of these microfunctions are carried out in the intervals between horizontal scan lines on the cathode ray tube display **24**.

In addition to the keys for moving the cursor location on the screen and the keys for the transfer of data, the following instruction keys are included on the keyboard with the function thereof listed on the right:

INSTRUCTION TABLE

KEY	FUNCTION

Clear Line	Clears the entire line from the cursor to the end of the line on the cathode ray tube screen.
Clear Message	Clears the screen from the cursor position to an end of message symbol (↑).
Clear Page	Clears the screen completely and causes the cursor to be put in the top lefthand position of the screen (home position).
Insert Character	Places a blank at the cursor position on the screen and moves all of the characters from the cursor position to the end of the line to the right by one position.
Delete Character	Deletes the character at the cursor position and causes each of the characters to the right of the deleted character to the end of the line to be moved one position to the left.
Insert Line	Moves all of the lines from the cursor line down one line from the cursor. The cursor line is left blank.
Delete Line	Deletes the line in which the cursor is located and moves all of the lines below the cursor up one line.
Enter	Places a new character at the location of the cursor (this instruction is carried out each time a new character is pressed on the keyboard 22).

Each of the above instructions is carried out by a plurality of the hereinabove specified microfunctions. In order to examine the operation of the edit control during the aforementioned instruction, it will be assumed that the display **24** has a capacity of only four lines of four characters. Thus, the fourth character in each line would be the last character in the line and the fourth line would be the last line on the display. In the examples hereinafter described, the positions will be denoted by the coordinates of the position on the screen. For example, the character in the first or top line and at the first or leftmost position on the line will be designated as being in position **1—1**. Similarly, if a character position for the location of the third line, second character position is denoted, the designation would be **3–2**.

In Chart I hereinbelow, the "Clear Line" instruction is shown prior to execution and after execution.

CHART I

CLEAR LINE INSTRUCTION

A B C D E F G H I J̄ ↑ L M N O P	A B C D E I J̄ ↑ L M N O P
Display Prior to Instruction	Display After Instruction Executed

It can be seen that the display prior to "Clear Line" instruction has the characters "A," "B," "C" and "D" in the first line and "E," "F," "G," "H" in the second line, "I," "J," "↑," "L" in the third line and "M," "N," "O," "P" in the fourth line. The character "↑" represents the end of message character and the line underneath the character "F" in the second line, second character position (**2—2**) represents the position of the cursor. Thus, the cursor is in the position on the screen at which the character "F" is located.

The Clear Line instruction is carried out by the following microfunctions during the sequence states associated therewith:

SEQUENCE STATE	MICROFUNCTION
S0	C→TC
S1	SET RECYCLE
S4B	BLANK→T
	TEST (Does C = **4**)
S5	T→RM
S5B	WRITE
S6	INCX
If C = **4**, proceed to **S7**, otherwise recycle	
S7	TC→C
	END EDIT

The sequence of the microfunctions above is best understood by referring to FIG. **3** wherein it can be seen that the instruction register **70** receives the Clear Line instruction when the Clear Line instruction key is pressed on the keyboard **22**. This causes the decoder **74** to provide a positive signal on the edit line **106** which corresponds to the Clear Line instruction. The line **106** which corresponds to the Clear Line instruction stays high until such time as the Clear Line instruction has been completely executed and the instruction register reset.

Referring now to FIG. **4**, it can be seen that OR gate **160** is enabled by the high signal on one of the lines **106** thereby causing flip-flop **162** to be set as soon as the next horizontal line of the scan raster of video display **22** is ended. Upon the reception by the sequence shift register **76** of the next sequence clock pulse a 1 is placed in stage **S0** of the shift register. During the time that flip-flop **S0** is set, one of the gates **200** in FIG. **7** of the microprogram function generator is enabled as a result of the high signal on the output line of flip-flop **S0**, the reception of the clock pulse CKA and the signal on the edit line **106** corresponding to the Clear Line instruction which causes the execution of the microfunction C → TC.

As set forth above, this microfunction causes the position of the cursor which is recorded in the cursor X and cursor Y counters to be recorded in the temporary cursor X store and Y store **84** and **86**, respectively. Upon the next sequence clock pulse, the 1 in stage **S0** is shifted to flip-flop stage **S1**.

As the "1" is shifted out of shift register stage **S0**, the recycle flip-flop **180** is set thereby causing the signal on the recycle line **188** to go high. The 1 is then shifted during the next three clock pulses from shift register stage **S1** to **S4**. During the period that the 1 is in shift register stages **S2** and **S3**, no further microfunctions take place. During the period that the 1 is in shift register stage **S4** and during the high portion of the clock pulse CKB, the T register **90** is set to a code representative of a blank (BLANK → T). Also during the time that a 1 is in shift register stage **S4**, one of the gates **186** (FIG. **5**) which is connected to the output of the gate **220** (FIG. **8**) of the limit detector **92** is enabled if the code for the last line position or the end of the line is detected. That is, in the preferred embodiment of the system, 40 characters are provided on a line. Accordingly, since the characters are numbered 0 to 39, if the code 39 is provided to gate **220**, then a limit signal would be provided to one of the gates **186**, which is also responsive to the fourth stage of the shift register **76**, and the edit line corresponding to the Clear Line instruction which would enable the gate **186**. However, if the limit is not reached, in other words, in this example, if the cursor X counter is not at 4, the gate **186** associated with this microfunction is not enabled and the recycle flip-flop **180** is not reset.

Accordingly, line **188** of the flip-flop **180** remains at a high voltage level. When the 1 in shift register stage **S4** is shifted to stage **S5**, the microfunction T → RM is carried out which effectively puts a blank code in the input register to the refresh memory.

When the clock pulse B goes high during the period that the 1 is in the flip-flop stage **S5**, the "Write" instruction is emitted by the microprogram function generator **78** and causes the blank to be written into the location in the refresh memory **2—2**. Accordingly, the screen goes blank at the position **2—2** at which the character F was originally shown. When the sixth

clock pulse causes the 1 to be shifted from S5 to S6, the function INCX is carried out which causes the cursor X counter **80** to be incremented and thus moves the cursor to the position 2—3. The seventh clock pulse does not cause the 1 in flip-flop stage S6 to be shifted into flip-flop S7 because of the fact that AND gate **174** cannot be enabled due to the recycle line **190** remaining low in potential. The 1 is then shifted out of the sequence register **76** and the edit control remains inoperative until the end of the next horizontal scan raster line.

At the end of the next horizontal scan raster line, the synchronizing clock pulse is again provided on lines **166** to flip-flops **162** and AND gate **192**. (FIGS. **4** and **5**) The trigger input to flip-flop **162** does not change the state of flip-flop **162** because the line **170** to the reset remains high thereby inhibiting the flip-flop from being switched to the set state as long as the recycle line **188** to OR gate **176** remains high.

However, the synchronizing clock pulse to line **166** to the AND gate **192** causes the flip-flop S2 of the shift register **76** to be set upon the next sequence clock pulse. This effectively puts a 1 in shift register stage S2. Thus, the sequence states S0 and S1 are eliminated from the next cycle of the sequence since the 1 is already in state S2 where it is shifted from S2 through S6. The microfucntions which were executed during the first sequence of the sequencer shift register **76** is thus repeated with the exception of the microfunctions which are initiated during the sequences of S0 and S1. Thus, when the 1 is shifted into the flip-flop stage S4, a blank code is again set into the T register **90** (BLANK → T). Also, during the sequence S4, the limit AND gates **186** are enabled to test whether the cursor X counter has the end of line code therein as determined by the gate **220** of the limit detector **92**. Since the X counter **80** was set to 3 during the previous sequence, the test for a 4 in the cursor X counter **80** fails. Accordingly, flip-flop **180** remains in the set state causing the recycle line **188** to remain high.

During the sequence state S5, the code for a blank space in the T register is transferred to the input means for the refresh memory. During the substate B of the S5 state of the sequencer, the "Rite" instruction is provided by the microprogram function generator to cause the blank space to be written into the portion of the refresh memory which formerly stored the character code for the letter "G." Thus, the screen is blank where the G formerly was displayed.

During the sequence state S6, the cursor X counter **80** is again incremented by the microprogram function generator **78** causing the cursor X counter to be incremented to the number 4. The cursor Y counter remains at 2.

Thus, it can be seen that the first two sequences of the sequence shift register **76** cause first the F to be blank and then the G to be blank. It should also be noted that since the recycle line **190** remains low, the next clock pulse causes the 1 in the shift register stage S6 to be shifted out thereby ending the operation of the edit control until the end of the next horizontal scan raster line.

As soon as the next horizontal scan raster line is completed, the synchronizing clock **166** again causes gate **192** to be enabled which in turn causes the shift register stage S2 to be set to the 1 state. Thus, in the next sequence states S2 through S6, the H in line 2 of the display is set to blank. However, during state S4B, the test for the 4 in the cursor X counter, causes the enablement of one of the gates **186** which causes the flip-flop **180** to be reset and thereby cause the output line **190** to go high thereby causing AND gate **174** to be enabled. Thus, after the 1 is in flip-flop state S6, the next sequence clock causes the 1 to be shifted into flip-flop stage S7. During sequence state S7, the cursor X counter and cursor Y counter are preset to the count presently in the temporary cursor X store and Y store **84** and **86**, respectively. Also, a signal is provided from the microprogram function generator to the instruction register **70** which is reset thereby ending the editing signal from the decoder **74** on one of the lines **106**.

It should be noted that since in the preferred embodiment of this invention, the characters that are displayed on each line of the display is 40, the recycle sequences would have continued until a 39 had been placed in the cursor X counter **80**. Even though the character positions 4 through 39 were blank, the editing instruction continues to the end of the line. This entire operation is completed during the retrace times between each of the horizontal lines on the video scan raster. It can, therefore, be seen that the entire instruction is executed in less time than is required to complete one video scan raster.

When the "Clear Page" instruction is provided by pressing the Clear Page instruction key in the keyboard **22**, the entire screen is made blank. The following is a schematic description of the operation of the editing control during the sequences S0 through S7. Where a sequence has been left out, it will be assumed that no editing microfunctions have taken place during the sequence state:

SEQUENCE MICRO-STATE	FUNCTION	DESCRIPTION
S0	RCX, RCY	Resets the cursor X and Y counters to 1-1.
S1	SET RECYCLE	Recycle flip-flop **180** is set making output line **188** high.
S4B	BLANK→T	The code for a blank space is placed in T register as in "Clear Line" instruction.
	TEST	A test to see if CX equals "4" and CY equals "4". (The end of line and last line counts and therefore the last position on the screen.)
S5	T→RM	The contents of the T register are then transferred to the input of the refresh memory.
S5B	WRITE	The contents of the input to memory are placed in the position set into the position addressed by the cursor and cursor Y counters.
S6	INCX	The cursor X counter is incremented.
	If CX = 4 and CY = 4, proceed to S7, if not, recycle	
S7	RCX, RCY	The cursor X and cursor Y counters are reset to 1-1 thereby causing the cursor to be left in the upper left corner of the screen at the end of the instruction.
	END EDIT	Reset the instruction register to cause an end of the "Clear Page" instruction.

It can be seen from the above sequence that during the first sequence of the shift register **76**, the cursor X and Y counters are set to the position 1—1. During state S1, the flip-flop **180** is set to cause output line **188** to go high. During state S4B, the character code for a blank is put in the T register and the test is made to determine whether the counters **80** and **82** have the end of line and last line codes therein. If they do not, the sequence will be recycled starting with a 1 placed in sequence state S2 and thus only the states S4B through S6 will be repeated. If, however, the recycle is initiated by reason of a 4 being in each of the cursor X and cursor Y counters, then a recycling will not take place and rather state S7 is enabled which causes the end of the edit.

It should be noted that the cursor X counter recycles after the last character position on a line. That is, if position 4 is the highest position that the cursor X counter counts to, it recycles to 1 and causes a carry to be put into the cursor Y counter which automatically increments the cursor Y counter. Thus, after the position 1-4 has been placed in the cursor counters, an incrementing of the cursor X counter causes the count in the cursor counters to be 2-1 and so on until the count

reaches 4—4 at which time the recycle flip-flop **180** is reset to cause the end of the cycle.

It can, therefore, be seen that in each of the sequences of the Clear Page instruction, one of the character spaces is blanked in the order of left to right and then down through the next line. At the end of the edit instruction, the cursor counter is reset to cause the cursor to be put in the home position in the upper left hand corner of the screen.

The Clear Message instruction clears the screen from the cursor position to the end of message symbol. Chart II which is provided hereinbelow illustrates the display prior to instruction and the display after the Clear Message instruction has been executed.

CHART II

CLEAR MESSAGE INSTRUCTION

A B C D E F G H I J ↑ L M N O P Display Prior to Instruction	A B C D E L M N O P Display After Instruction Executed

The Clear Message instruction is carried out in exactly the same sequence as the Clear Line instruction with the exception that in the state and substate S4B, the test in the Clear Message is made to determine whether the end of message character is located at the position in the refresh memory at which the cursor counter is presently addressing. As soon as the end of message character (↑) is reached, the recycle flip-flop is reset thereby preventing further recycling and termination of the Clear Message instruction.

The "Insert Character" instruction causes a blank space to be inserted at the position of the cursor with each of the characters from the cursor to the right of the line moved one position to the right. This instruction is best understood in connection with Chart III hereinbelow which shows the display prior to the Insert Character instruction and the display after the Insert Character instruction has been executed.

CHART III

INSERT CHARACTER INSTRUCTION

A B C D E F G H I J ↑ L M N O P Display Prior to Instruction	A B C D E F G I J ↑ L M N O P Display After Instruction Executed

It should be noted that the H which is originally at the end of the line is lost due to the fact that there are no more character positions remaining after the fourth position on the second line. In the preferred embodiment where the display includes 40 character positions, the character H is located to the right of the G as illustrated after the instruction is executed.

The sequence of operations in the Insert Character instruction are as follows:

SEQUENCE MICRO-STATE	FUNCTION	DESCRIPTION
S0	C→TC	The cursor address is stored in the temporary cursor store.
S1	BLANK→ID	The code for a blank is set into the ID register.
	SET	The recycle flip-flop is set.
	RECYCLE	
S3	READ	A memory cycle is started whereby the address of the refresh memory designated by the count in the cursor X and cursor Y counter is accessed.
S4B	RM→T	The contents of the addressed portion of the refresh memory are transferred to the T register.
	TEST	The cursor X counter is checked to determine whether the cursor is at the end of the line.
S5	ID→RM	The contents of the ID register is provided to the input of the refresh memory.
S5B	WRITE	The blank code which has just been transferred from the ID register is written into the location of the refresh memory at which the cursor X and Y counters are positioned.
S6A	INCX	The cursor X counter is incremented.
S6B	T→ID	The contents of the T register are preset into the ID register.
	If during the test, it is determined that the cursor X counter indicates the end of the line, proceed to S7, otherwise recycle	
S7	TC→C	The contents of the temporary cursor X and Y store are preset into the cursor X and cursor Y counters.

In order to carry out the Insert Character instruction on the display shown in Chart III prior to the Insert Character instruction, the following events occur: During the first sequence, the cursor address 2—2 is stored in the temporary cursor store. The ID register is set to the blank code and the recycle flip-flop set. The character F at the position 2—2 in the refresh memory is then transferred to the T register. A test is made to determine whether the end of line code is in the cursor X counter. Since it is not, the recycle flip-flop remains set. The contents of the ID register which is the code for a blank is then sent to the memory position 2—2 causing a blank to appear where the F formerly appeared. The cursor X counter is then incremented causing the cursor address to be 2–3. The F in the T register is then transferred to the ID register.

During the second sequence, the operations taking place during states S0 and S1 are eliminated due to the fact that the 1 is placed directly into flip-flop S2. Accordingly, the contents of the refresh memory at cursor address 2–3 are read into the T register. Thus, a code for the character G is set in the T register. A test is then made to determine whether a 4 is in the cursor X counter. Since it is not, the recycle flip-flop remains set. The code for the character F in the ID register is then transferred to the refresh memory to the position specified in the cursor X counter, namely, position 2–3. The cursor X counter is then incremented so that the cursor address is 2–4. The code for the character G in the T register is then transferred to the ID register. During the third sequence, again, as in all recycles, the state S2 is initiated first so that the microfunctions appearing during states S0 and S1 are again eliminated. The contents of the refresh memory at address 2–4, namely, the code for the character H, are read into the T register. A test is then made to determine whether the 4 appears in the cursor X counter. Since it does, the recycle flip-flop is reset. The character code for the letter G in the ID register is then transferred to the refresh memory to position 2–4. The cursor X counter is then incremented causing the address threin to go to 3–1. The character code for the letter H in the T register is then transferred to the ID register. (Since

this is the last recycle, the H is lost since the H is not transferred to the refresh memory.)

Since the recycle flip-flop has been reset, the microfunctions of state S7 of the sequence are caused to be executed thereby causing the original cursor address which has been stored in the temporary cursor store to be preset into the cursor X and cursor Y counters. Also the instruction register is reset thereby ending the edit instruction.

The "Delete Line" instruction causes the line in which the cursor is located to be completely deleted and each of the lines of characters therebelow are moved up one line.

As seen in chart IV hereinbelow, the Delete Line instruction is performed in two parts.

CHART IV

DELETE LINE INSTRUCTION

A B C D E F G H I J ↑L M N O P Display Prior to Instruction	A B C D I J ↑L M N O P A B C D Display After First Part of Instruction	A B C D I J ↑ L M N O P Display After Instruction Executed

As seen in Chart IV, the first part of the instruction causes the deletion of the second line of the display and each of the succeeding lines are copied up one line. It should be noted that the top line has been copied into the fourth line of the display after the first part of the editing instruction. Thus, the second part of the instruction is a Clear Line instruction which starts from the cursor position 4–1 and proceeds to cursor position 4—4.

The sequence of instructions to carry out the first edit of the Delete Line instruction is as follows:

SEQUENCE MICRO-STATE	FUNCTION	DESCRIPTION
S0	C→TC	Initial cursor address is stored in the temporary cursor storage register.
S1	RCX	The cursor X counter is reset to "1".
S2	INCY	The cursor address in the cursor Y counter is moved down one line.
S3B	READ	The memory cycle is initiated to access the contents of the refresh memory at the position specified in the cursor X and cursor Y counter.
S4A	DECY	Cursor Y counter is decremented to move the cursor up one line.
S4B	RM→T	The contents of the refresh memory position that was accessed is fed to the T register.
S5	T→RM	The character code in the T register is fed to the input for the refresh memory.
S5B	WRITE	The character code at the input of the refresh memory is read into the refresh memory specified by the cursor X and cursor Y counter.
S6A	TEST	The test is made for the end of line and last line in the cursor X and cursor Y counters.
S6B	INCX	The cursor is moved one position to the right.
	If cursor counter equals 4–4, proceed to S7, otherwise recycle	
S7	END EDIT 1	The first edit which comprises the "Delete Line" instruction is then terminated by resetting the appropriate portion of the instruction register and the second edit is initiated by setting a different portion of the instruction register.

The second edit which forms the second portion of the Delete Line instruction is similar to the Clear Line instruction except that during state S0, instead of presetting the temporary cursor storage to the count presently in the cursor X and cursor Y counters, the cursor X counter is reset. Thus, the cursor counter is set to 4–1 and the 2—2 which is stored in the temporary cursor storage register is placed in the cursor X and cursor Y counter during state S7 of the last sequence of the instruction.

The Enter Instruction causes a new character to be placed at the location of the cursor.

CHART V

ENTER INSTRUCTION

A B C D E F G H I J ↑L M N O P Display Prior to Instruction	A B C D E R G H I J ↑L M N O P Display After Letter "R" is Entered

Thus, as seen in Chart V, with the display as shown prior to instruction and the letter R key on the keyboard 22 depressed, the letter R is placed in the position of the cursor with the previous character located thereat erased. It should also be noted that the cursor is moved to the next position.

As soon as a character key is depressed on the character keyboard, the code for the character is placed directly into the ID register 94. The depression of the character key also causes the "Enter" instruction to be placed in the instruction register 70 which causes the following sequence of microfunctions in the edit control:

SEQUENCE MICRO-STATE	FUNCTION	DESCRIPTION
S4	ID→T	The "R" in the ID register is sent to the T register.
S5	R RM	The "R" in the T register is then sent to the input of the refresh memory.
S5B	WRITE	The character "R" is then written into the refresh memory at the position specified by the cursor X and cursor Y counters.
S6A	INCX	The cursor X counter is incremented by "1" moving the cursor X position one space to the right.
S6B	TEST	The test is unconditional thereby causing the recycle flip-flop to be reset and thereby cause the AND gate 194 to be enabled to pass the "1" from flip-flop stage S6 to S7.
S7	END EDIT	The instruction register is reset to end the edit signal.

It can therefore be seen that the entering of data is similar to the operation of a typewriter with the cursor moved to the next position so that the next character key depressed causes the character to be written at the position of the cursor.

The results of a Delete Character instruction are shown in Chart VI.

CHART VI

DELETE CHARACTER INSTRUCTION

A B C D E F G H I J↑L M N O P Display Prior to Instruction	A B C D E G H I J↑L M N O P Display After Instruction Executed

The sequence of operations for the delete character instruction is as follows:

SEQUENCE STATE	MICROFUNCTION
S0	C→CT
S2	INCX
S3B	READ
S4A	DECX
S4B	RM→T TEST (for C = 4)
S5A	BLANK→T only if C = 4
S5	T→RM
S5B	WRITE
S6A	INCX
If C = 4, proceed to S7, otherwise recycle	
S7	TC→C END EDIT

In the Delete Character instruction, the character at the cursor is removed and the next character placed in its position during the first sequence. In the next sequences, the remaining characters are moved one position to the left. In the last sequence, the last character in the line is moved to the next to last position and the last position is filled with a blank space.

The results of an Insert Line instruction are shown in Chart VII. The insert line instruction also requires two edit sequences to complete the instruction.

CHART VII

INSERT LINE INSTRUCTION

A B C D A B C D E F G H E F G H I J L E F G H M N O P I J L Display Prior to Instruction Display After Execution of First Edit	A B C D E F G H I J L Display after Complete Instruction Executed

During the first edit sequences, each of the lines from the cursor line down are copied in the line below. It should be noted that the bottom line is lost. The first sequence of operations is, therefore, as follows:

SEQUENCE STATE	MICROFUNCTION
S0	C→TC
S1	RCX, SET CY TO 4
S2	DECY
S3B	READ
S4A	INCY
S4B	RM→T
S5A	T→RM
S5B	DECY only if C = 4 WRITE
S6A	INCX (inhibit a carry to CY)
S6B	TEST (for TCY = CY)
If TCY = CY, proceed to S7, otherwise recycle	
S7	END EDIT 1

As soon as edit 1 of the Insert Line instruction is completed, a Clear Line instruction is initiated. Since the cursor is at 2–1 prior to the Clear Line instruction, the second line is cleared with the cursor remaining at position 2–1.

It can therefore be seen that a new and improved edit control for a video display terminal has been provided. The edit control includes a plurality of temporary storage registers to enable the data in the refresh memory to be moved around in order to accomplish instructions which are otherwise impossible on existing video display terminals without requiring extensive erasing of the character display and consequent reinsertion of the necessary data.

It can also be seen that the execution of the instructions is substantially immediate and is carried out during the retrace time between each of the horizontal scan lines of the video scan raster.

It should be understood that the instructions which have been described above are exemplary only and that the edit control enables various other instructions with only a single depression of a key on the keyboard being required to execute these instructions.

Without further elaboration, the foregoing will so fully illustrate my invention that others may, by applying current or future knowledge, readily adapt the same for use under various conditions of service.

What is claimed as the invention is:

1. In a video display terminal having input means for receiving data and instructions, a refresh memory responsive to said input means, and a cathode ray tube, controlled by the contents of said refresh memory, tube for displaying data in accordance with the data and the position of said data in said refresh memory inserted in said refresh memory by said input means by providing an alpha-numeric representation of said data in a scan raster comprised of a plurality of parallel scan lines, editing means, said editing means being responsive to said input means and controlling the data and location of the data in said refresh memory and being initiated at the end of each of said scan lines to immediately enter said data into said memory and carry out said instructions for changing the location of data in said memory, said editing means being operable only during the time between the successive scan lines on said cathode ray tube.

2. The video display terminal of claim 1 wherein the periods between scan lines are divided into a plurality of intervals.

3. The invention of claim 2 wherein said intervals are defined by a sequencing means having a plurality of output lines which are sequentially energized during sequential portions of said interval between said scan raster lines.

4. The invention of claim 3 wherein a plurality of said periods between scan lines are required, said sequencing means being operated for one cycle during each of said periods.

5. The invention of claim 3 wherein said sequencing means comprises a shift register and said output lines comprise the output lines of the stages of said shift register.

6. The invention of claim 5 wherein means are provided for inserting an enabling bit into the first stage of said shift register only during a first cycle, and means for inserting an enabling bit into an intermediate stage of said shift register in subsequent cycles.

7. The invention of claim 6 wherein said enabling bit is shifted from the stage in which it is inserted to the next to last stage of the shift register in all cycles but the last wherein said enabling bit is also shifted into the last stage of said shift register.

8. An edit control for a video display terminal having a memory which stores the character codes for each character displayed and a character display which is controlled by said memory to display in alpha-numeric form the characters stored in said memory in accordance with the location of said character codes in said memory, said edit control including

address means comprising a counter for access to said memory, the count of said counter controlling the location in said memory to which and from which a character code may be transferred, temporary storage means connected to said address means for storing the address in said address means at the start of an editing operation, temporary character code storage means, and means for initiating the transfer of data between said temporary character code storage means and said memory, said means for initiating transfer of data controlling the count of said counter so that the characters in said display may be entered, removed, or moved in accordance with the entering, removing, or moving of said character codes in said memory.

9. The edit control of claim 8 and further including a sequence means, said sequence means being connected to said means for initiating transfer of data so that a plurality of said data transfers may be sequentially initiated.

10. The edit control of claim 9 and further including an instruction register for storing an instruction until it has been executed by said edit control, said means for initiating transfer of data also being responsive to said instruction to provide a predetermined set of transfers in accordance with the instruction in said instruction register.

11. The edit control of claim 8 wherein said sequencing means comprises a shift register, said shift register having an output line for each stage thereof, said output lines enabling said means for initiating transfer of data.

12. The edit control of claim 11 wherein said means for initiating transfer of data is responsive to said instruction and said plurality of output lines to provide sequences of data transfers in accordance with the signals provided on said output lines.

13. An edit control responsive to instructions for manipulating the contents of the display of a video display terminal having a memory which stores the character codes for each character display position and a character display which is controlled by said memory, and address means for accessing said memory, said edit control further including means for generating a plurality of sequence states responsive to instructions received by said edit control, and means responsive to a first portion of said sequence states to execute at least one microinstruction, to a second portion of said sequence states to execute at least one microinstruction, and to a third portion of said sequence states to execute at least one microinstruction, said means for generating a plurality of sequence states generating said first and second portion of said sequence states upon receipt by said edit control of an instruction, test means responsive to the address means and said memory to detect a predetermined condition to enable the generation of said third portion of said sequence states otherwise said testing means enables the regeneration of said second portion of said sequence states until said predetermined condition is detected, and transfer means, said microinstructions controlling said address means and said transfer means for entering, removing and moving character codes in said memory.

* * * * *