(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) Publication number: **0 069 518**
B1

(12) EUROPEAN PATENT SPECIFICATION

(45) Date of publication of patent specification: **12.08.87**

(51) Int. Cl.⁴: **G 09 G 1/16**

(21) Application number: **82303343.6**

(22) Date of filing: **25.06.82**

(54) Raster scan video display terminal.

(30) Priority: **06.07.81 US 280619**

(43) Date of publication of application:
12.01.83 Bulletin 83/02

(45) Publication of the grant of the patent:
12.08.87 Bulletin 87/33

(84) Designated Contracting States:
BE DE FR GB IT NL SE

(56) References cited:
EP-A-0 031 011
US-A-4 249 172

IBM TECHNICAL DISCLOSURE BULLETIN, vol. 21, no. 11, April 1979, pages 4332-4333, New York, US; H.C. TANNER: "Linking algorithm with segmentation"

ELECTRO/81 CONFERENCE RECORD, vol. 6, 7th-9th April 1981, pages 1-11, New York, US

(73) Proprietor: **DATA GENERAL CORPORATION**
4400 Computer Drive
Westboro Massachusetts 01580 (US)

(72) Inventor: **Hunt, Glenn E.**
7401 Haddick Circle
Austin Texas 78745 (US)
Inventor: **Alexander, Michael C.**
1818 South Lakeshore Boulevard
Austin Texas 78741 (US)
Inventor: **Gittins, Robert S.**
4904 Cabot Street
Austin Texas 78741 (US)
Inventor: **Lozano, Gerald L.**
3407 Sanderling Trail
Austin Texas 78746 (US)

(74) Representative: **Pears, David Ashley et al**
REDDIE & GROSE 16 Theobalds Road
London WC1X 8PL (GB)

EP 0 069 518 B1

Courier Press, Leamington Spa, England.

Description

The present invention relates to a raster scan video display terminal, as set forth in the introductory part of claim 1.

The image on a CRT is generated by using an electron beam to stimulate selected areas of a phosphorescent material located on the inside of the CRT screen. The scanning of the CRT face is accomplished by deflecting the electron beam relatively rapidly in one direction, usually horizontal, and relatively slowly in a second direction, usually vertical. The phosphorescent material on the screen is continuous, but the screen can be considered to consist of a large number of generally horizontal, parallel "raster lines" or lines of displayed information. As the beam scans along a raster line, the information about the level of stimulation to be given a particular area on the raster line is updated at fixed intervals in accordance with a clock pulse or "dot clock". Therefore, each raster line can be further considered to be a series of discrete segments or "dots" which are individually stimulatable by the electron beam.

The electron beam normally performs 50 or 60 "frames" or complete scans of the CRT screen per second, depending on the external electrical power available. From the viewpoint of an observer facing the screen the beam begins a frame at the left side of the top raster line of the CRT and moves substantially horizontally along the line to the right side of the screen stimulating each dot to the appropriate level to create the desired image. The beam then performs a horizontal retrace to the left side of the next lower raster line and again begins to scan horizontally to the right. This continues until the beam reaches the right side of the lowest raster line, at which time a vertical retrace is performed during which the beam moves back to the beginning of the top raster line to begin the next frame. No information is displayed during either horizontal or vertical retrace.

A problem in the prior art is the extremely high work load of the CPU which can result from user changes to the display. In the prior art, data to be displayed is commonly stored in sequential memory locations in terminal memory. The first character to be displayed (i.e. the leftmost character of the top row) is not necessarily located in the first memory location and is typically indicated by a "top of page" pointer. The leftmost character of displayed row 2 is stored in the memory location immediately following the rightmost character of row 1, and so on, with the rightmost character of the last row being the end of the "string". If, for example, a character is to be inserted into the display and therefore inserted into the "string" of characters sequentially stored in memory, the addresses of all characters following the insertion must be changed to reflect their new position in the string. If the insertion occurs near the top of the screen, a substantial amount of processor work must be performed to change the memory locations of all following characters. To complete the operation during vertical retrace requires the terminal to have a very fast CPU and memory. To allow the operation to continue over multiple frames presents the terminal user with a visible "ripple" effect as the memory is updated.

A retailed prior art problem is the high processor workload resulting from the method of performing vertical or horizontal scrolling. The avoid display degradation or delays, prior art terminals which provide scrolling capability must use a processor capable of performing the data movements required under the prior art method.

A partial solution to these problems is known from EP 0 031 011 which describes a terminal according to the introductory part of claim 1, with a separate table of addresses for the rows of characters, whereby such operations as scrolling require manipulation of the table only, rather than the stored rows of characters. However, the addresses in the table have to be maintained correctly, in sequential order as they are accessed sequentially.

It is known from IBM Technical Disclosure Bulletin Vol. 21 No. 11 April 1979 pp. 4332, 4333 to supplement each row of characters with address pointers to the next rows, which thus form a linked list.

The object of the present invention is to provide a terminal which simplifies screen manipulation even further and opens up possibilities for much more flexible screen handling, including smooth scrolling.

The terminal according to the invention is defined in the characterising part of claim 1.

The display information for a row need only be stored if the content of the display information for that row is changed. This applies equally to the row description information. Updating can take place during vertical retrace.

The display information can be retrieved by repeating for each row the steps of reading from the description information in the memory the memory address of the information to be displayed on the row, reading from memory the information to be displayed on the row, and reading from the memory the pointer address of the description information pertaining to the next row.

The method of retrieval and display of information can involve repeating for each row the steps of transferring the information to be displayed to buffers while simultaneously displaying the first raster line of the row, displaying the remaining raster.

The row description information for each row can include the information related to the address of the first character to be displayed on the row; the first raster line to be displayed within the row; the number of raster lines to be displayed with the row; control information related to vertical synchronization, end of frame identification and blanking of the display during vertical retrace; and the pointer address of the row description information for the next row to be displayed.

Such row description information allows

smooth vertical scrolling by changing the first raster line and number of raster lines to be displayed. Moreover, the row description information allows horizontal scrolling without requiring changes to stored character information, simply by changing the first character addressed in the row description informations.

The row description information also allows for control of vertical synchronization, display density, display blanking identification of the end of the frame.

The invention will now be described in more detail, by way of example, with reference to the accompanying drawings, in which:

Fig. 1 is a block diagram of a CRT terminal embodying the present invention.

Fig. 2 is a block diagram of the Video Control Logic and Video Character Generation Logic of Fig. 1.

Fig. 3 is a block diagram illustrating a possible structuring of display data.

Fig. 3A is a block diagram illustrating another possible structuring of display data.

Fig. 4 is a block diagram illustrating a technique for upward vertical display scrolling.

Fig. 5 is a block diagram illustrating a technique for downward vertical display scrolling.

For clarity of presenting and illustrating the invention, a terminal having specific parameters will be used as the basis for discussion, but it should be understood that the invention is not limited to a single specific set of numbers or dimensions. Obviously, many terminal parameters will depend on such factors as CRT size, semiconductor operating limitations and monitor performance characteristics. Therefore, the following discussion will assume a terminal having 288 total displayed scan lines. The displayed scan lines allow 24 displayed horizontal "rows" of characters of 12 scan lines each. Within each row, the displayed character occupies scan lines 2 through 10 (i.e., character height is 9 scan lines). If 22 scan line times occur during vertical retrace while no information is being displayed, the terminal can be viewed as cyclicly performing 310 (288 + 22) horizontal scans per vertical scan line.

Referring to Fig. 1 an overflow of the internal logic of an intelligent video display terminal is shown. CPU 100 interfaces with Character Data Bus 191 via bidirectional buffer 110, System Data Bus 192 via bidirectional buffer 111, Attribute Data Bus 193 via bidirectional buffer 112 and Downline Loadable Character Bus 194 via bidirectional buffer 113. Buffers 110 and 112 each interface a different address space of RAM (Random Access Memory) 150 to CPU 100. Data are transferred over Character Data Bus 191 to Address Latches 300, Ram 150, Video Control Logic 200 and Video Character Generation Logic 250. Data related to the various system devices with which the terminal may interface (e.g. keyboard, printer) is carried via System Data Bus 192 to and from System Devices Logic 130. Data specifying the attributes (e.g. dim, blink, underscore, inverse) of the characters to be displayed are transferred via Attribute Data Bus 193 to RAM 150 and Video Character Generation Logic 250. Downline Loadable Character Bus 194 allows terminal users to transfer their own unique characters to CPU 100 for display. Address bus 195 is connected to Address Latches 300, Decoders 120, System Devices Logic 130, Buffers 140 and RAM 150.

Decoder Logic 120 contains logic to decode the information on Address Bus 195 to determine which, if any, system device is being addressed. Buffers 140 provide the appropriate TTL to MOS interface, as required by RAM 150 and some elements of System Devices 130 (e.g. ROM's).

Video Control Logic 200 is connected to CPU 100, Address Latches 300, Buffer 110, Line Buffers 160, Video Timing Logic 400, Latch 170, Ram 150, Video Character Generation Logic 250 and CRT Monitor 180. Video Character Generation Logic 250 is connected to Buffers 110 and 112, Line Buffers 160, Video Timing 400, Latch 170, and RAM 150. CPU 100 is connected via System Device Logic 130 to the host computer (not shown) external to the terminal and communicates with the host over System Data Bus 192.

Referring now to Fig. 2, a more detailed schematic of Video Control Logic 200, Line Buffers 160 and Video Character Generation Logic 250 is shown. Video Control Logic 200 generates the horizontal synchronization signal for the monitor drive electronics; provides synchronization between CPU 100 and RAM 150; controls the transfer of information from RAM 150 to Character Generation Logic 250 and Line Buffers 161—164; and prevents access by CPU 100 to RAM 150 during transfers of display information (described below) to Line Counter 203, Raster Counter 254, Status Latch 202, and Line Buffers 161—164. CPU 100 controls Video Control Logic 200 only by means of a discrete halt line, which is used during initial setup of the display information after a hardware restart.

Character Generator Logic 250 receives character and attribute data from data buses 191 and 193 and from Line Buffers 161—164, control information from Video Control Logic 200, and timing signals from Timing Logic 400 (not shown in Fig. 2). Character Generation Logic 250 combines the character, attribute and control information and generates the dot pattern for transmission to monitor 180.

State Counter 201 counts the character time periods during each scan line and provides the character count to State Machine 210. Line Counter 203 receives information from Character Data Bus 191 and notifies State Machine 210 when the first scan line of each character row is being displayed. Status Latch 202, under control of State Machine 210, provides an interrupt signal to State Machine 210, character format information to Latch 220, a vertical sync signal to Latch 170 and a vertical blanking signal to Attribute Encoding Logic 263. State Machine 210 provides control signals to CPU 100, Adress Latches 300

and State Counter 201. State Machine 210 also supplies the horizontal synchronization signal to Latch 220.

Character Latch 251 receives character, data from bus 191 on the first scan line of each character row. This data is supplied simultaneously to Line Buffers 161 and 162 and Character Latches 252. Similarly Attribute Latch 261 receives attribute data from bus 193 during the first scan line of each character row and supplies it simultaneously to Line Buffers 163 and 164 and Attribute Latch 262. Raster Counter 254, under State Machine 210 control, receives raster address information from bus 191. This information is supplied to Character Generator 253, which also receives the character information from Latches 252. Similarly, Raster Counter 254 is connected to Attribute Encoding Logic 263, as is Attribute Latch 262.

The output of Character Generator 253 is provided to Shift Registers 271. The output of Attribute Encoding Logic 263 is provided to latch 270, two outputs of which are supplied to Gates 280 where they are combined with the outputs of Shift Registers 271. A third output of Latch 270 is supplied directly to Latch 170 along with the vertical synchronization signal from Status Latch 202 and the output of Gates 280.

For purposes of illustration, assume again the typical terminal having 288 displayed scan lines 22 horizontal scan cycles required for vertical retrace. These 288 lines are equivalent to 24 character rows of 12 scan lines each, but because of the smooth scrolling capability discussed below, during some vertical scans the top and bottom rows in the scroll "window" will be only partially displayed. This requires CPU 100 to maintain 25 rows of character information in RAM 150.

This terminal embodiment allocates 8K bytes of RAM 150 for storage of attribute and character information. This memory space allows CPU 100 to store character and attribute information for 162 characters in RAM 150 for each of the 25 character rows.

During each vertical retrace period, CPU 100 will update and store the row information from which the display will be created during the next vertical scan. This row data (character and attribute) is organized on a row basis, rather than a screen basis. That is, each row of characters is stored in consecutive memory locations, but the rows are not arranged in any particular order. They are, instead, "linked" by means of RDB's (Row Descriptor Blocks), also assembled by CPU 100.

Each character row has associated with it one RDB consisting of five 8-bit bytes of information. The first, or Status byte contains the information about row format (81 or 135 character line), end of frame, vertical synchronization and vertical blanking. The second, or scroll, byte contains information about which scan line in the character row will be the first to be displayed and how many scan lines of the character row will be displayed. This information enables "smooth" vertical scrolling by allowing less than the entire character row to be displayed during a frame. The third and fourth bytes contain the starting address in RAM 150 of the 81 or 135 characters (depending on the format identified in the Status byte) to be displayed on that row. This information enables horizontal scrolling of the display within the 162 characters stored in RAM 150 for that row by simply changing the address in RDB bytes three and four. No change to character information in RAM 150 is required. The fifth, or Next RDB, byte is a pointer to the next RDB. That is, it contains the address of the next RDB to be used. Since the 8 bits of the Next RDB byte allow only 256 addresses, the RDB's are placed in the lowest memory locations in RAM 150. With five bytes per RDB, up to 51 possible RDBs can be used.

Advantages of RDB usage can now be clearly understood. For example, significant reductions in CPU work load and required memory speed can be realized. Since the display information is stored in RAM 150 by rows, rather than in a continuous sequence for the entire screen, the CPU is no longer required to move lengthy strings of character and attribute information for each character change. Rather, all character rows which do not require modification during a vertical retrace need not be moved in memory. Only the memory locations for the row being changed are affected.

Moving displayed rows on the screen requires only that the RDB's be "relinked". That is, that the Next RDB bytes be changed. With 24 rows of character information, there will be 24 linked row RDB's. In addition, three vertical retrace RDB's are inserted after the last displayed row. These retrace RDB's do not display any information and cover a total of 22 scan lines (i.e. the retrace period). The last retrace RDB points to the RDB of the first displayed row. The complete RDB list will contain either 27 RDB's (24 + 3), if 24 rows are completely displayed, or 28 RDB's (25 + 3) if scrolling is underway and two rows are only partially displayed. A possible linking situation is shown in Fig. 3.

For simplicity of design, RDB1 is chosen to always reside in the lowest memory location. The RDB's in Fig. 3 are shown in the order of displayed character rows. That is, bytes three and four of RDB1 contain the starting memory address of displayed row 1 and the Next RDB byte (byte five in this embodiment) contains the address of RDB5. Bytes three and four of RDB5 contain the starting memory address of displayed row 2 and the Next RDB byte contains the address of RDB3. The remaining RDB's are similarly linked. RDB28 in this example is the last character row and, therefore, the Next RDB byte of RDB28 contains the address of the first of three vertical retrace RDB's. The third vertical retrace RDB points back to RDB1.

Now, assume the terminal user wishes to remove displayed row 2. Rather than the CPU having to revise and store a substantial part of the

entire screen in memory, only the row associated with RDB5 and three RDB bytes need to be changed. Specifically, in this example, the Next RDB byte of RDB1 is changed to the address of RDB5, the Next RDB byte of RDB28 is changed to the address of RDB5, and the Next RDB byte of RDB5 is changed to the address of RDB22. RDB3 is now the RDB of the last character row and previous rows 3—25 have been "moved up". This situation is illustrated in Fig. 3A.

Also, smooth scrolling either up or down can be performed for all displayed rows on the screen or a subset thereof selected by a terminal user. As stated above, the scroll byte of each RDB contains information about which of the 12 scan lines in the row will be the first to be displayed and how many of the lines will be displayed. Smooth scrolling can be accomplished by modifying the scroll bytes of the RDB's associated with the top and bottom character rows in the scroll area and relinking the RDB's as required.

Fig. 4 presents an illustrative example of RDB activity related to vertical scrolling at a rate of one scan line every frame. Of course, the particular RDB reference numbers and RDB linkage order shown is of no particular importance beyond this example.

The numbers inside the RDB boxes in Fig. 4 indicate the data in the scroll byte of that RDB. Specifically, the total number of scan lines of that character row to be displayed and the starting scan line within the row are given. For example, looking at RDB7 in Fig. 4, 12/1 indicates that all 12 scan lines of the character row will be displayed starting with the first (i.e. top) line.

Each of the columns in Fig. 4 shows a segment of the 'list" of linked RDB's. Looking first at Frame n, assume upward vertical scrolling of the screen area now occupied by the character rows associated with RDB12 and RDB9, i.e. a scrolling space 24 scan lines high, is about to begin. During Frame n there are a total of 27 RDB's linked as described earlier. As shown for Frame n + 1, however, during a scrolling operation two character rows will normally be only partially displayed, requiring that an additional RDB be linked into the RDB list. Of course, the total number of displayed scan lines in the scroll area is constant (24, in this example).

During the vertical retrace between Frame n and Frame n + 1, CPU 100 will load the appropriate locations of RAM 150 with the information for the new RDB (in this example RDB 20) and with the character and attribute information for the row now associated with that RDB. In addition, the scroll byte of RDB 12 must be modified to indicate that only 11 scan lines, beginning with line 2, will be displayed and the Next RDB byte of RDB9 must be modified to point to RDB 20 instead of RDB 11. The Next RDB byte of RDB 20 will contain the address of RDB 11.

As indicated in Fig. 4, only the top line of the RDB 20 character row will be displayed during Frame n + 1. The total number of displayed scan lines in the scroll area has stayed constant at 24.

During the vertical retrace between Frame n + 1 and Frame n + 2, no RDB relinking is required and no change to the character or attribute information stored in RAM 150 is required. Only changes to the scroll byte of the RDB of the top row currently being displayed in the scroll area (in this example, RDB12) and the RDB of the bottom now currently being displayed in the scroll area (in this example, RDB 20) are necessary. Specifically, the scroll byte of RDB12 must be modified such that only 10 scan lines, beginning with line 3, are displayed. Similarly RDB 20 is modified such that now the top two scan lines of its associated character row are displayed during frame n + 2.

Modification of the scroll byte of RDB12 and RDB20 continues in this manner until the vertical retrace prior to Frame n + 12. Since the RDB12 character row has now been completely scrolled "off" the screen RDB12 is removed from the RDB linked sequence and the Next RDB byte of RDB7 is modified to point to RDB9. To the user, the display has scrolled upward by one character row. At the next vertical retrace, a new RDB (in this example, RDB 12) is linked into the list and the process described above for Frame n + 1 is repeated.

At a typical monitor operating rate of 60 frames per second, this technique will result in a scrolling rate of 60 scan lines (i.e. five character rows) per second. Other scrolling rates can be achieved. For example, a 10 row per second rate can be obtained by modifying the scroll bytes by two scan lines per frame rather than one as in Fig. 4.

Fig. 5 presents an illustrative example of downward scrolling at two scan lines per frame. The relinking and scroll byte modification is similar to that described above for upward scrolling except that the new RDB is linked in at above the other RDB's of the rows in the scroll area rather than after. Since scrolling is being performed at 2 scan lines per frame, the row associated with the bottom row in the scroll area (RDB9 in this example) will be completely removed from the screen in 5 frames rather than 10, as in the example of Fig. 4.

This terminal also has the capability for horizontal scrolling of displayed information. Horizontal scrolling is accomplished by changing the starting memory address (RDB bytes three and four) for that row. As mentioned earlier, RAM 150 contains 162 characters for each row, of which only an 81 or 135 character subset is displayed at any one time. Changing the contents of RDB bytes three and four causes a different subset of the 162 characters available in RAM 150 to be loaded into Line Buffers 161—164 for display. The actual character data in RAM 150 therefore need not be changed during the horizontal scrolling process.

It can be seen that the format for each row is independent of the format of any other row and is determined by the format information stored in the Status byte (byte one in this implementation) of the RDB for that row. Any combination of the display formats can, therefore, be set up by CPU 100 during vertical retrace. The actions necessary

to progress from character row to character row during vertical scan are controlled by Video Control Logic 200. In summary, starting during the last scan line of each row, Video Control Logic 200, will request CPU 100 to relinquish bus control; will obtain status, raster and address information from the next RDB; will transfer the character and attribute information for the row to Line Buffers 161—164; and will release CPU 100 prior to the end of the first scan line of the following row. This sequence of events continues to repeat during vertical retrace, even though no information is being displayed. The three vertical retrace RDB's, as stated earlier, are designed to maintain proper operation and synchronization during the retrace time period until the next vertical scan begins. The particular character information in Line Buffers 161—164 during vertical retrace is irrelevant since the blanking bit of the Status byte of the three vertical retrace RDB's is set to preclude display of any information during this period.

Claims

1. A raster scan video display terminal comprising memory means (150) for storing a plurality of rows of characters to be displayed, a buffer (160) for storing one row of characters during the line scans in which it is displayed, a dot matrix character generator (253) responsive to the buffer contents and line counter means (203) to provide a video signal for display of the characters, means (200, 300) for transferring characters row by row from the memory means to the buffer during each frame of the raster scan, and a stored table of addresses of the first characters, characterised in that the memory means (150) also store, at addresses independent of the addresses pertaining to those of the rows of characters, a set of description blocks (RDB) one for each row of characters, each description block storing information comprising the address of the first character to be displayed in the corresponding row and a pointer address for the description block for the next row to be displayed, and in that the means for transferring (200, 300) respond in respect of each row to the first character address in the description block pointed to by the description block for the preceding row.

2. A terminal according to claim 1, characterised in that the memory means (150) is updated during the vertical retrace intervals of the scan.

3. A terminal according to claim 2, characterised in that only rows which change are updated.

4. A terminal according to claim 2 or 3, characterised in that a description block (RDB) is only updated when its information changes.

5. A terminal according to any of claims 1 to 4, characterised in that attribute information is also stored in the memory means (150) in association with each stored character.

6. A terminal according to any of claims 1 to 5, characterised in that each row description block (RDB) also includes an indication of the number of characters in a complete row.

7. A terminal according to any of claims 1 to 6, characterised in that each row description block (RDB) also includes an indication of the first raster line of the character row to be displayed and of the number of raster lines of the character row to be displayed.

8. A terminal according to claim 7, characterised in that smooth vertical scrolling is effected by modifying the row description blocks (RDB) pertaining to the top and bottom displayed rows, namely by progressively modifying the number of the first line to be displayed and complementarily modifying the number of raster lines to be displayed, during the course of a plurality of frames.

9. A terminal according to any of claims 1 to 8, characterised in that each row description block (RDB) also includes an indication of the end of the frame, vertical synchronization and blanking during the vertical retrace.

10. A terminal according to any of claims 1 to 9, characterised by register means (202, 203, 254) for storing the row description information pertaining to the current row, and in that row description information from the block (RDB) pointed to by the contents of the register means is transferred to the register means in the horizontal retrace interval following the last line scan of each row.

11. A terminal according to any of claims 1 to 10, characterised in that characters are stored in strings larger than can be displayed in a row and in that horizontal scrolling is effected without changing the stored strings by altering the first character address in the row description blocks (RDB).

12. A terminal according to any of claims 1 to 11, characterised in that the row description blocks (RDB) are stored in a set of contiguous address blocks in the memory means (150) and the block to block pointers are one-byte pointers only.

Patentansprüche

1. Nach dem Rasterverfahren arbeitendes Videosichtgerät, enthaltend eine Speichereinrichtung (150) zum Speichern einer Vielzahl von sichtbar zu machenden Zeichenreihen, einen Zwischenspeicher (160) zum Speichern einer Zeichenreihe während der Zeilenabtastungen, in denen diese sichtbar gemacht wird, einen Punktmatrixzeichengenerator (253), der auf den Inhalt des Zwischenspeichers anspricht, eine Zeilenzähleinrichtung (203) zur Erzeugung eines Videosignals zur Sichtbarmachung der Zeichen, Übertragungsmittel (200, 300) zur Übertragung von Zeichen Reihe für Reihe aus der Speichereinrichtung zum Zwischenspeicher während jedes Bildes der Rasterabtastung, und eine gespeicherte Tabelle der Adressen der ersten Zeichen der Zeichenreihen, dadurch gekennzeichnet, daß die

Speichereinrichtung (150) bei Adressen, die unabhängig von den zu den Zeichenreihen gehörenden Adressen sind, auch je einen Satz von Reihenbeschreibungsblöcken (RDB) für jede Zeichenreihe Speichert, wobei jeder Reihenbeschreibungsblock Informationen speichert, die die Adresse des ersten sichtbar zu machenden Zeichens in der entsprechenden Reihe und eine Zeigeradresse für den Beschreibungsblock der nächsten sichtbar zu machenden Reihe speichert, und daß die Übertragungsmittel (200, 300) bezüglich jeder Reihe auf die Adressen der ersten Zeichen in dem Reihenbeschreibungsblock ansprechen, auf den durch den Reihenbeschreibungsblock für die vorhergehende Reihe gezeigt wird.

2. Videosichtgerät nach Anspruch 1, dadurch gekennzeichnet, daß die Speichereinrichtung (150) während der vertikalen Rücklaufintervalle der Abtastung aktualisiert wird.

3. Videosichtgerät nach Anspruch 2, dadurch gekennzeichnet, daß nur Reihen, die sich ändern, aktualisiert werden.

4. Videosichtgerät nach Anspruch 2 oder 3, dadurch gekennzeichnet, daß ein Beschreibungsblock (RDB) nur aktualisiert wird, wenn sich seine Information ändert.

5. Videosichtgerät nach einem der Ansprüche 1 bis 4, dadurch gekennzeichnet, daß in der Speichereinrichtung (150) in Verbindung mit jedem gespeicherten Zeichen auch Attributeninformationen gespeichert werden.

6. Videosichtgerät nach einem der Ansprüche 1 bis 5, dadurch gekennzeichnet, daß jeder Reihenbeschreibungsblock (RDB) auch eine Angabe über die Zahl der Zeichen in der vollständigen Reihe umfaßt.

7. Videosichtgerät nach einem der Ansprüche 1 bis 6, dadurch gekennzeichnet, daß jeder Reihenschreibungsblock (RDB) auch eine Angabe über die erste Rasterzeile der sichtbar zu machenden Zeichenreihe und über die Zahl der Rasterzeilen der sichtbar zu machenden Zeichenreihe enthält.

8. Videosichtgerät nach Anspruch 7, dadurch gekennzeichnet, daß durch Modifizierung der zu der obersten und untersten sichtbar gemachten Reihe gehörenden Reihenbeschreibungsblöcke ein sanftes Rollen erzeugt wird, indem im Verlauf einer Vielzahl von Bildern die Zahl der ersten sichtbar zu machenden Zeile fortschreitend und die Zahl der sichtbar zu machenden Rasterzeilen komplementär dazu modifiziert wird.

9. Videosichtgerät nach einem der Ansprüche 1 bis 8, dadurch gekennzeichnet, daß jeder Reihenbeschreibungsblock (RDB) auch eine Angabe über das Rasterende, die Vertikalsynchronisation und die Austastung während des vertikalen Rücklaufs enthält.

10. Videosichtgerät nach einem der Ansprüche 1 bis 9, dadurch gekennzeichnet, daß Registermittel (202, 203, 254) zum Speichern der zur laufenden Reihe gehörigen Reihenbeschreibungsinformationen vorgesehen sind und daß Reihenbeschreibungsinformationen von dem Block (RDB), der durch den Inhalt der Registermittel angezeigt wird, in die Registermittel während desjenigen horizontalen Rücklaufintervalls übertragen werden, das der letzten Zeilenabtastung jeder Reihe folgt.

11. Videosichtgerät nach einem der Ansprüche 1 bis 10, dadurch gekennzeichnet, daß die Zeichen in Zeichenketten gespeichert werden, die größer sind, als in einer Reihe sichtbar gemacht werden kann, und daß ein horizontales Rollen ohne Änderung der Gespeicherten Zeichenketten dadurch herbeigeführt wird, daß die erste Zeichenadresse in den Reihenbeschreibungsblöcken (RDB) geändert wird.

12. Videosichtgerät nach einem der Ansprüche 1 bis 11, dadurch gekennzeichnet, daß die Beschreibungsblöcke (RDB) in der Speichereinrichtung in einem Satz von angrenzenden Adressenblöcken gespeichert werden und die Block-zu-Block-Zeiger nur Ein-Byte-Zeiger sind.

Revendications

1. Terminal d'affichage vidéo à balayage par trame comprenant des moyens de mémoire (150) pour mémoriser une pluralité de rangées de caractères à afficher, un tampon (160) pour mémoriser une seule rangée de caractères pendant les balayages de lignes dans lesquels elle est affichée, un générateur de caractères à matrice de points (253) agissant en réponse au contenu du tampon et à des moyens compteurs de lignes (203) pour fournir un signal vidéo pour l'affichage des caractères, des moyens (200, 300) pour transférer les caractères rangée par rangée depuis les moyens de mémoire vers le tampon pendant chaque trame du balayage par trame, et une table mémorisée d'adresses des premiers caractères des rangées de caractères, caractérisé en ce que les moyens de mémoire (150) mémorisent également, à des adresses indépendantes des adresses appartenant à celles des rangées de caractères, un ensemble de blocs de description (RDB), un pour chaque rangée de caractères, chaque bloc de description mémorisant des informations comprenant l'adresse du premier caractère à afficher dans la rangée correspondante et une adresse de pointeur pour le bloc de description de la rangée suivante à afficher, et en ce que les moyens pour transférer (200, 300) répondent en relation avec chaque rangée à la première adresse de caractère dans le bloc de description sur lequel est détecté un pointage par le bloc de description de la rangée précédente.

2. Terminal selon la revendication 1, caractérisé en ce que les moyens de mémoire (150) sont mis à jour pendant les intervalles de retour vertical du balayage.

3. Terminal selon la revendication 2, caractérisé en ce que seulement des rangées qui changent sont mises à jour.

4. Terminal selon l'une des revendications 2 ou 3, caractérisé en ce qu'un bloc de description (RDB) est mis à jour seulement quand ses informations changent.

5. Terminal selon l'une quelconque des reven-

dications 1 à 4, caractérisé en ce que les informations d'attributs sont également mémorisées dans les moyens de mémoire (150) en association avec chaque caractère mémorisé.

6. Terminal selon l'une quelconque des revendications 1 à 5, caractérisé en ce que chaque bloc de description de rangée (RDB) comprend également une indication du nombre de caractères dans une rangée complète.

7. Terminal selon l'une quelconque des revendications 1 à 6, caractérisé en ce que chaque bloc de description de rangée (RDB) comprend également une indication de la première ligne de trame de la rangée de caractères à afficher et du nombre de ligne de trame de la rangée de caractères à afficher.

8. Terminal selon la revendication 7, caractérisé en ce qu'un défilement vertical lisse est effectué en modifiant les blocs de description de rangées (RDB) appartenant aux rangées affichées supérieure et inférieure, en particulier en modifiant progressivement le numéro de la première ligne à afficher et en modifiant de façon complémentaire le nombre des lignes de trame à afficher, au cours d'une pluralité de trames.

9. Terminal selon l'une quelconque des revendications 1 à 8, caractérisé en ce que chaque bloc de description de rangée (RDB) comprend également une indication de la fin de la trame, de la synchronisation verticale et de l'effacement pendant le retour vertical.

10. Terminal selon l'une quelconque des revendications 1 à 9, caractérisé par des moyens de registre (202, 203, 254) pour mémoriser des informations de description de rangée appartenant à la rangée en cours, et en ce que les informations de description de rangée en provenance du bloc (RDB) pointées par les contenus des moyens de registre sont transférées aux moyens de registre pendant l'intervalle de retour horizontal suivant le dernier balayage de ligne de chaque rangée.

11. Terminal selon l'une quelconque des revendications 1 à 10, caractérisé en ce que des caractères sont mémorisés selon des chaînes plus grandes que cela ne peut être affiché dans une rangée et en ce qu'un défilement horizontal est effectué sans modifier les chaînes mémorisées en modifiant la première adresse de caractère dans les blocs de description de rangée (RDB).

12. Terminal selon l'une quelconque des revendications 1 à 11, caractérisé en ce que les blocs de description de rangée (RDB) sont mémorisés selon un ensemble de blocs d'adresses contiguës dans les moyens de mémoire (150) et en ce que les pointeurs bloc à bloc sont des pointeurs à un multiplet seulement.

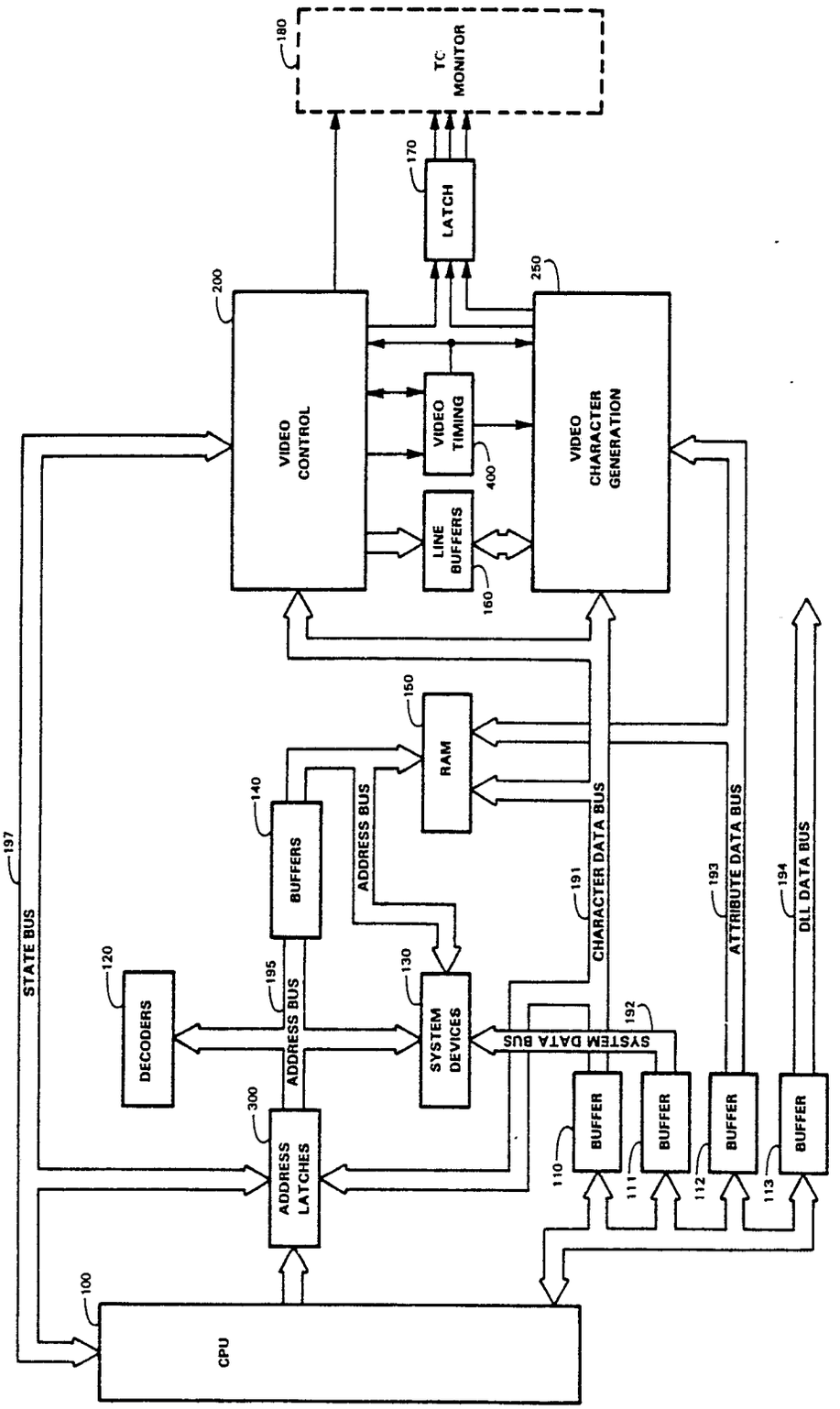
180
TO MONITOR
170
LATCH
200
VIDEO CONTROL
250
VIDEO CHARACTER GENERATION
VIDEO TIMING
400
LINE BUFFERS
160
150
RAM
140
BUFFERS
ADDRESS BUS
CHARACTER DATA BUS
ATTRIBUTE DATA BUS
DLL DATA BUS
191
193
194
197
STATE BUS
120
DECODERS
195
ADDRESS BUS
130
SYSTEM DEVICES
192
SYSTEM DATA BUS
300
ADDRESS LATCHES
BUFFER
110
BUFFER
111
BUFFER
112
BUFFER
113
100
CPU

FIG 1

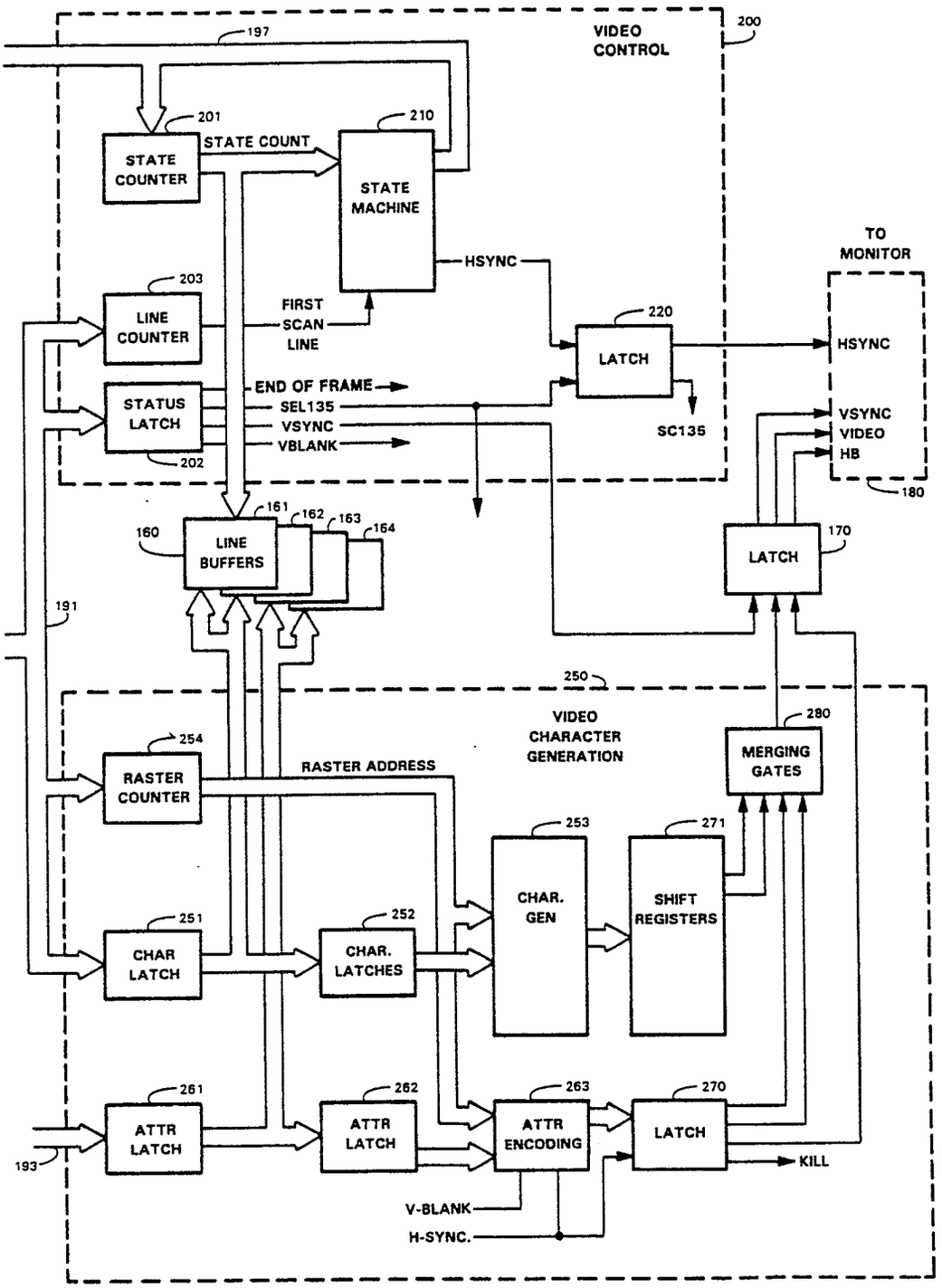
VIDEO CONTROL
200
197
201
STATE COUNTER
STATE COUNT
210
STATE MACHINE
HSYNC
TO MONITOR
203
LINE COUNTER
FIRST SCAN LINE
220
LATCH
HSYNC
END OF FRAME
SEL135
VSYNC
VBLANK
STATUS LATCH
202
SC135
VSYNC
VIDEO
HB
180
161
162
163
164
160
LINE BUFFERS
LATCH
170
191
250
VIDEO CHARACTER GENERATION
280
MERGING GATES
254
RASTER COUNTER
RASTER ADDRESS
253
271
CHAR. GEN
SHIFT REGISTERS
251
CHAR LATCH
252
CHAR. LATCHES
261
262
263
270
ATTR LATCH
ATTR LATCH
ATTR ENCODING
LATCH
193
KILL
V-BLANK
H-SYNC.

FIG 2

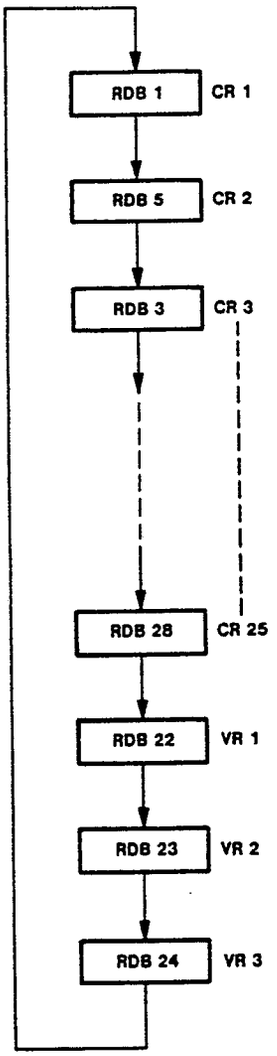
RDB 1
CR 1
RDB 5
CR 2
RDB 3
CR 3
RDB 28
CR 25
RDB 22
VR 1
RDB 23
VR 2
RDB 24
VR 3

FIG 3

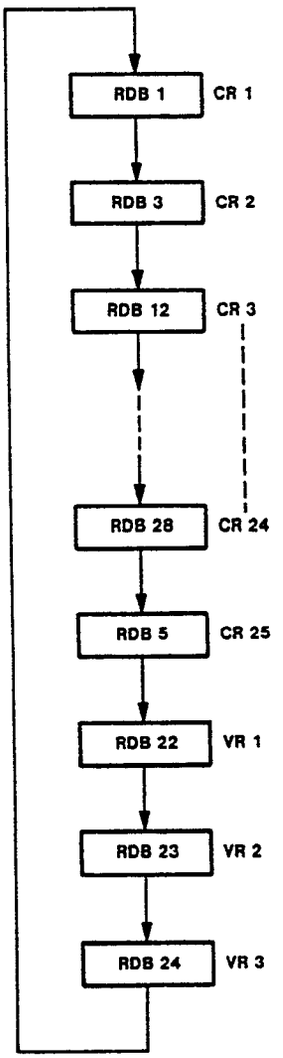
RDB 1
CR 1
RDB 3
CR 2
RDB 12
CR 3
RDB 28
CR 24
RDB 5
CR 25
RDB 22
VR 1
RDB 23
VR 2
RDB 24
VR 3

FIG 3A

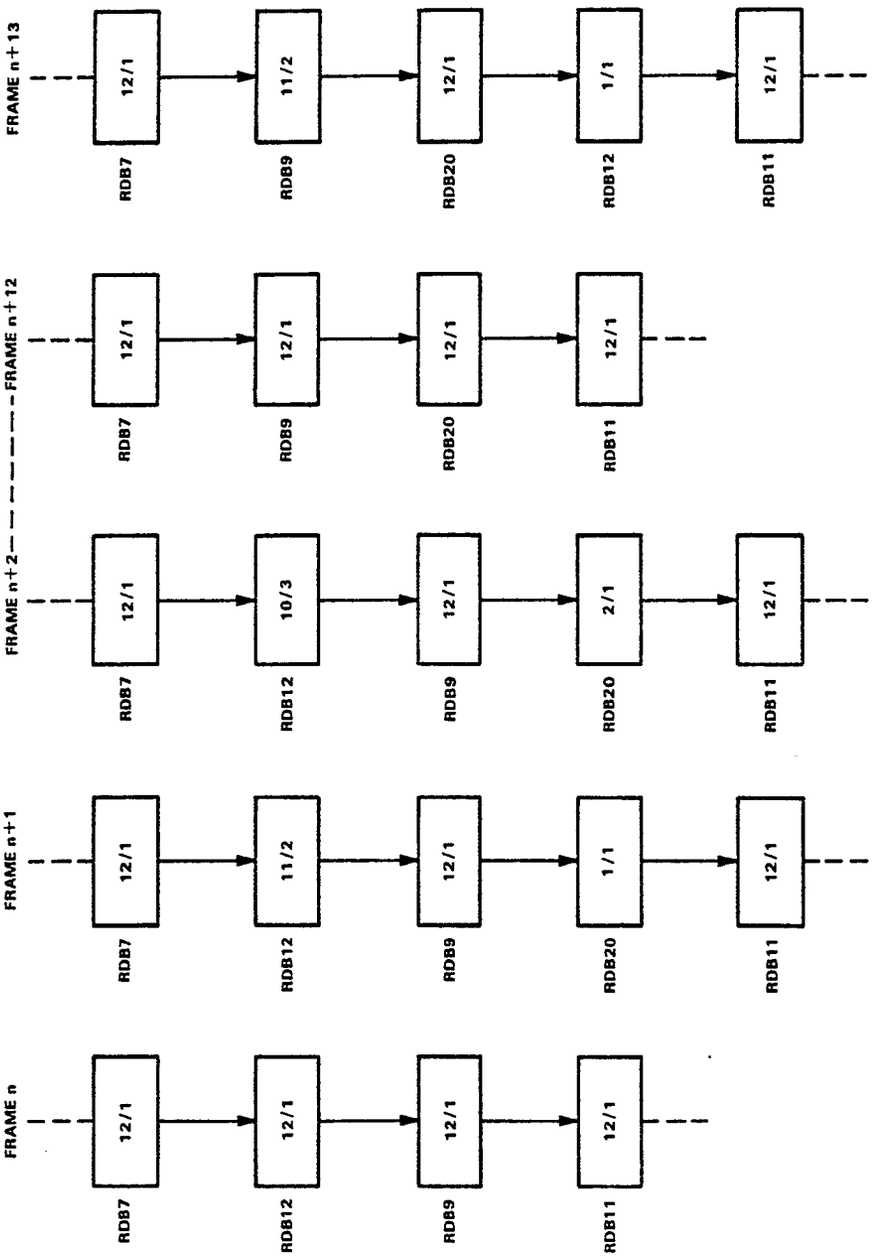
FRAME n+13
12/1
11/2
12/1
1/1
12/1
RDB7
RDB9
RDB20
RDB12
RDB11
FRAME n+12
12/1
12/1
12/1
12/1
RDB7
RDB9
RDB20
RDB11
FRAME n+2
12/1
10/3
12/1
2/1
12/1
RDB7
RDB12
RDB9
RDB20
RDB11
FRAME n+1
12/1
11/2
12/1
1/1
12/1
RDB7
RDB12
RDB9
RDB20
RDB11
FRAME n
12/1
12/1
12/1
12/1
RDB7
RDB12
RDB9
RDB11

FIG 4

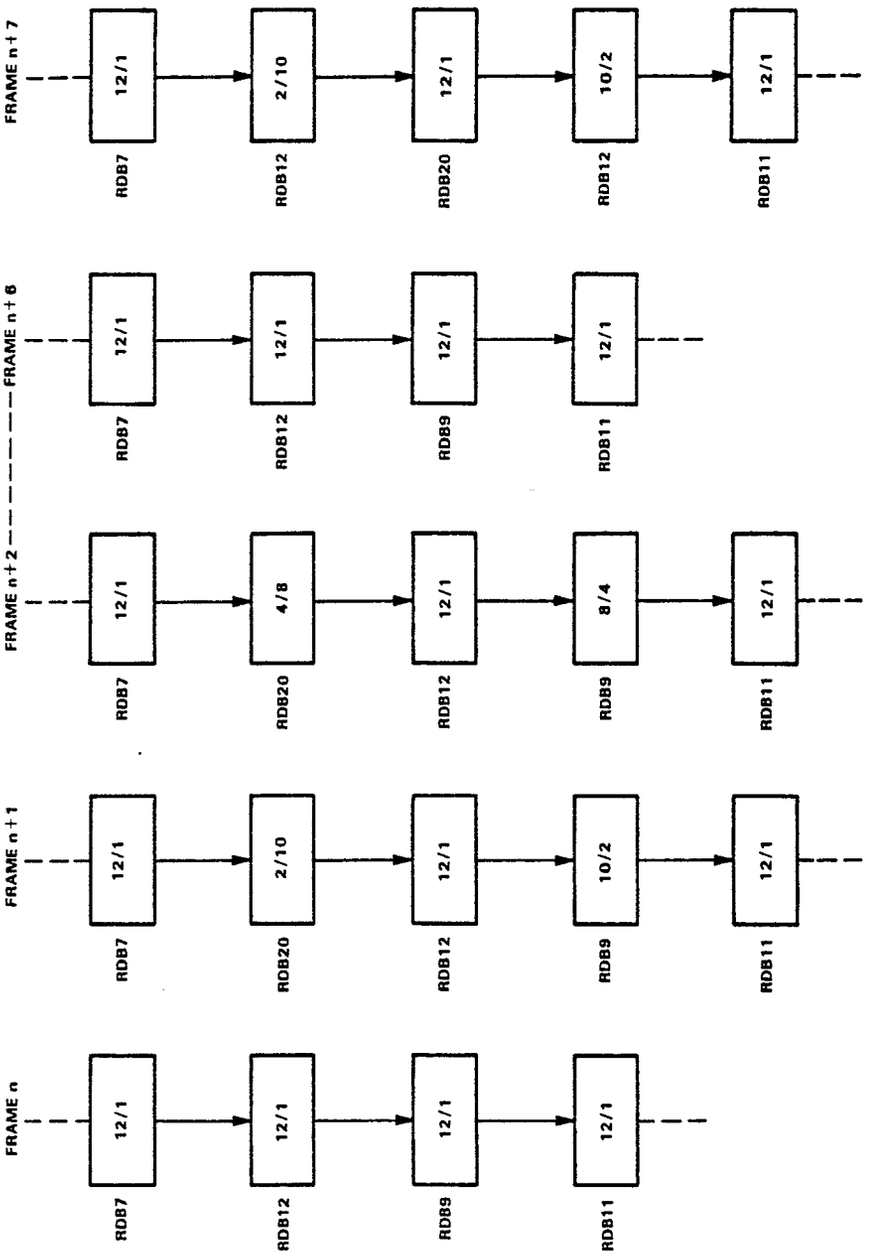
FRAME n+7
12/1
2/10
12/1
10/2
12/1
RDB7
RDB12
RDB20
RDB12
RDB11
FRAME n+6
12/1
12/1
12/1
12/1
RDB7
RDB12
RDB9
RDB11
FRAME n+2
12/1
4/8
12/1
8/4
12/1
RDB7
RDB20
RDB12
RDB9
RDB11
FRAME n+1
12/1
2/10
12/1
10/2
12/1
RDB7
RDB20
RDB12
RDB9
RDB11
FRAME n
12/1
12/1
12/1
12/1
RDB7
RDB12
RDB9
RDB11

FIG 5