Office de la Propriété Intellectuelle du Canada
Un organisme d'Industrie Canada

Canadian Intellectual Property Office
An agency of Industry Canada

CA 2684678 A1 2011/05/03

(21) 2 684 678

(12) DEMANDE DE BREVET CANADIEN
CANADIAN PATENT APPLICATION

(13) A1

(22) Date de dépôt/Filing Date: 2009/11/03

(41) Mise à la disp. pub./Open to Public Insp.: 2011/05/03

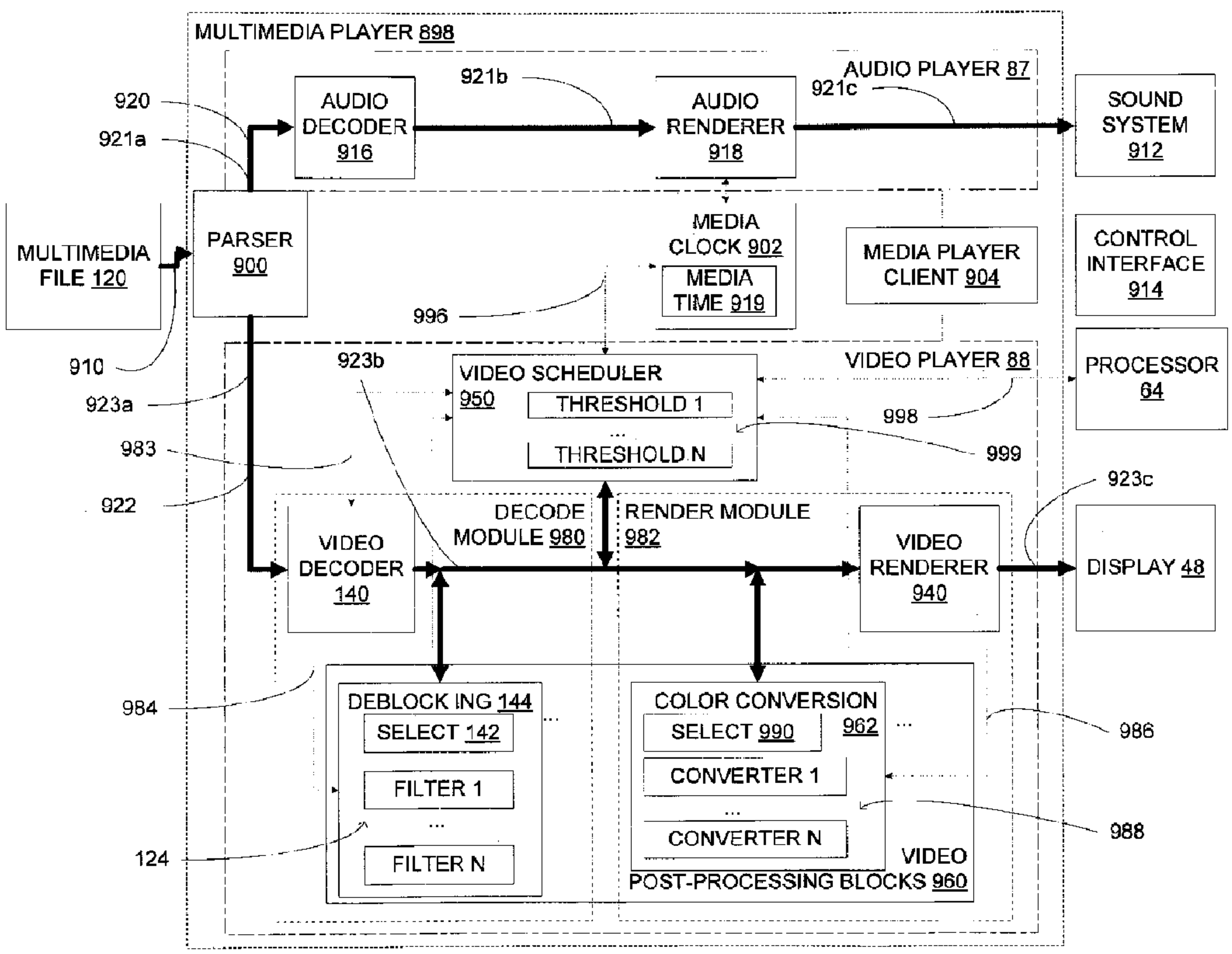
(51) Cl.Int./Int.Cl. *H04N 21/43* (2011.01), *H04N 5/775* (2006.01), *H04W 4/00* (2009.01), *H04W 88/02* (2009.01)

(71) Demandeur/Applicant:
RESEARCH IN MOTION LIMITED, CA

(72) Inventeurs/Inventors:
GLAZNEV, ALEXANDER, CA;
MAK-FAN, DAVID, CA;
SMALL, AARON, CA

(74) Agent: RIDOUT & MAYBEE LLP

(54) Titre : SYSTEME ET METHODE APPLICABLES AU POST-TRAITEMENT DYNAMIQUE D'UN APPPAREIL MOBILE
(54) Title: SYSTEM AND METHOD FOR DYNAMIC POST-PROCESSING ON A MOBILE DEVICE



(57) Abrégé/Abstract:
A method of operating a multimedia player is provided. The method includes decoding an audio stream of the multimedia player and rendering the decoded audio stream in the multimedia player, updating the media time of the media player with an audio timestamp of the rendered audio stream as the audio stream is rendered, and, while decoding and rendering the audio stream, decoding a video stream and checking the media clock to determine if a video timestamp of the decoded video stream is within a threshold of the media clock time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.

ABSTRACT

A method of operating a multimedia player is provided. The method includes decoding an audio stream of the multimedia player and rendering the decoded audio stream in the multimedia player, updating the media time of the media player with an audio timestamp of the rendered audio stream as the audio stream is rendered, and, while decoding and rendering the audio stream, decoding a video stream and checking the media clock to determine if a video timestamp of the decoded video stream is within a threshold of the media clock time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.

SYSTEM AND METHOD FOR DYNAMIC POST-PROCESSING ON A MOBILE DEVICE

TECHNICAL FIELD:

[0001] The following relates to systems and methods for applying post-processing to a video stream.

BACKGROUND

[0002] A computing device, such as a mobile device, uses resources, such as a processor, to perform tasks. Each task inherently consumes a certain percentage of the overall resources of the device. It is well known that mobile devices generally have fewer resources than, for example, personal computers (PCs). Many tasks, often referred to as non-interactive tasks, are fixed tasks that are scheduled by a scheduling algorithm. Other tasks, often referred to as interactive tasks, in some way relate to recent input/output (I/O) traffic or user related tasks, such as user input or user directed output. The scheduling algorithm typically aims to schedule interactive tasks for optimal low latency and non-interactive tasks for optimal throughput.

[0003] An example of a non-interactive task is video decoding, which is done in the background, and an example of an interactive task is a keystroke or status bar update that is to be presented to a user on the display of the mobile device. The video content expected to be played on a mobile device often pushes the capabilities of mobile devices. Video decoding can be part of decoding a multimedia file including both audio content and video content. Devices for playing multimedia content are often referred to as media players. In some circumstances, the mobile device cannot decode video sufficiently quickly to maintain synchronization with playback of the audio content. This can provide a poor viewing experience to the user.

BRIEF DESCRIPTION OF THE DRAWINGS

[0004] Example embodiments will now be described with reference to the appended drawings wherein:

[0005] Figure 1 is a schematic diagram of a mobile device and a display screen therefor.

[0006] Figure 2 is a schematic diagram of another mobile device and a display screen therefor.

[0007] Figure 3 is a schematic block diagram of components of the mobile device of any or both of Figures 1 and 2.

[0008] Figure 4 is a schematic block diagram of the memory shown in Figure 3.

[0009] Figure 5 is a screen shot of a home screen for the mobile device of any or both of Figures 1 and 2.

[0010] Figure 6 is a screen shot of a video player interface illustrating a blocking artefact.

[0011] Figure 7 is a screen shot of the video player interface of Figure 6 illustrating another blocking artefact.

[0012] Figure 8 is a screen shot of the video player interface of Figure 6 illustrating yet another blocking artefact.

[0013] Figure 9 is a schematic block diagram of the media player shown in Figure 4.

[0014] Figure 10 is a schematic block diagram showing an embodiment for streaming media content.

[0015] Figure 11 is a schematic diagram of an example multimedia file shown in Figure 4.

[0016] Figure 12 is an example audio stream in the media player of Figure 4 for the multimedia file of Figure 4.

[0017] Figure 13 is an example video stream in the media player of Figure 4 for the multimedia file of Figure 4.

[0018] Figures 14-18 are example illustrations of different video lag scenarios for the audio stream and video stream of Figures 12 and 13 during processing by the multimedia player 898.

[0019] Figure 19 is a flow diagram showing an example procedure for adaptive post-processing of a video stream based on video lag.

[0020] Figure 20 is a flow diagram showing an example frame discarding post-processing adaptation for the procedure of Figure 19.

[0021] Figure 21 is a flow diagram showing an example degrading of color space conversion quality post-processing adaptation for the procedure of Figure 19.

[0022] Figure 22 is a flow diagram showing an example degrading of deblocking post-processing adaptation for the procedure of Figure 19.

[0023] Figure 23 is a flow diagram showing an example method of multiple threshold and multiple post-processing adaptation.

[0024] Figure 24 is a flow diagram showing an example method of multiple threshold and multiple post-processing adaptation based on processor usage.

DETAILED DESCRIPTION OF THE DRAWINGS

[0025] Example mobile devices and methods performed thereby are now described for dynamic post-processing of a video stream. The method is performed according to video processing lag from a rendered audio stream.

[0026] Referring now to Figures 1 and 2, one embodiment of a mobile device 10a is shown in Figure 1, and another embodiment of a mobile device 10b is shown in Figure 2. It will be appreciated that the numeral "10" will hereinafter refer to any mobile device 10, including the embodiments 10a and 10b. It will also be appreciated that a similar numbering convention may be used for other general features common between Figures 1 and 2 such as a display 12, a positioning device 14, and a cancel or escape button 16.

[0027] The mobile device 10a shown in Figure 1 comprises a display 12a and the cursor or view positioning device 14 shown in this embodiment is a positioning wheel 14a. Positioning device 14 may serve as another input member and is both rotatable to provide selection inputs to the processor 64, (see Figure 3) and can also be pressed in a direction generally toward housing to provide another selection input to the processor 64. As shown in Figure 3, the processor 64 may be a microprocessor. The methods and structure herein can be applied to a wide range of processors 64; however, a mobile device 10 will typically utilize a microprocessor as the processor 64. The display 12 may include a selection cursor 18 that depicts generally where the next input or selection will be received. The selection cursor 18 may comprise a box, alteration of an icon or any combination of features that enable the user to identify the currently chosen icon or item. The mobile device 10a in Figure 1 also comprises an escape or cancel button 16a and a keyboard 20. In this example, the keyboard 20 is disposed on the front face of the mobile device housing and positioning device 14 and cancel button 16a are disposed at the side of the housing to enable a user to manoeuvre the positioning wheel 16a while holding the mobile device 10 in one hand. The keyboard 20 is in this embodiment a standard QWERTY keyboard.

[0028] The mobile device 10b shown in Figure 2 comprises a display 12b and the positioning device 14 in this embodiment is a trackball 14b. Trackball 14b permits multi-directional positioning of the selection cursor 18 such that the selection cursor 18 can be moved in an upward direction, in a downward direction and, if desired and/or permitted, in any diagonal direction. The trackball 14b is preferably situated on the front face of a housing for mobile device 10b as shown in Figure 2 to enable a user to manoeuvre the trackball 14b while holding the mobile device 10b in one hand. The trackball 14b may serve as another input member (in addition to a directional or positioning member) to provide selection inputs to the processor 64 and can preferably be pressed in a direction towards the housing of the mobile device 10b to provide such a selection input.

[0029] The mobile device 10b also comprises a menu or option button 24 that loads a menu or list of options on display 12b when pressed, and a cancel or escape button 16b to exit, "go back" or otherwise escape from a feature, option, selection or display. The mobile device 10b as illustrated in Figure 2, comprises a reduced QWERTY keyboard 22. In this embodiment, the keyboard 22, positioning device 14, escape button 16b and menu button 24 are disposed on a front face of a mobile device housing.

[0030] The reduced QWERTY keyboard 22 comprises a plurality of multi-functional keys and corresponding indicia including keys associated with alphabetic characters corresponding to a QWERTY array of letters A to Z and an overlaid numeric phone key arrangement. The plurality of keys that comprise alphabetic and/or numeric characters total fewer than twenty-six (26). In the embodiment shown, the number of keys that comprise alphabetic and numeric characters is fourteen (14). In this embodiment, the total number of keys, including other functional keys, is twenty (20). The plurality of keys may comprise four rows and five columns of keys, with the four rows comprising in order a first, second, third and fourth row, and the five columns comprising in order a first, second, third, fourth, and fifth column. The QWERTY array of letters is associated with three of the four rows and the numeric phone key arrangement is associated with each of the four rows.

[0031] The numeric phone key arrangement is associated with three of the five columns. Specifically, the numeric phone key arrangement may be associated with the second, third and fourth columns. The numeric phone key arrangement may alternatively be associated with keys in the first, second, third, and fourth rows, with keys in the first row including a number "1" in the second column, a number "2" in the third column, and a number "3" in the fourth column. The numeric phone keys associated with keys in the second row include a number "4" in the second column, a number "5" in the third column, and a number "6" in the fourth column. The numeric phone keys associated with keys in the third row include a number "7" in the second column, a number "8" in the third column, and a number "9" in the fourth column. The numeric phone keys associated with keys in the fourth row may include a "*" in the second column, a number "0" in the third column, and a "#" in the fourth column.

[0032] The physical keyboard may also include a function associated with at least one of the plurality of keys. The fourth row of keys may include an "alt" function in the first column, a "next" function in the second column, a "space" function in the third column, a "shift" function in the fourth column, and a "return/enter" function in the fifth column.

[0033] The first row of five keys may comprise keys corresponding in order to letters "QW", "ER", "TY", "UI", and "OP". The second row of five keys may comprise keys corresponding in order to letters "AS", "DF", "GH", "JK", and "L". The third row of five keys may comprise keys corresponding in order to letters "ZX", "CV", "BN", and "M".

[0034] It will be appreciated that for the mobile device 10, a wide range of one or more positioning or cursor/view positioning mechanisms such as a touch pad, a joystick button, a mouse, a touchscreen, set of arrow keys, a tablet, an accelerometer (for sensing orientation and/or movements of the mobile device 10 etc.), or other whether presently known or unknown may be employed. Similarly, any variation of keyboard 20, 22 may be used. It will also be appreciated that the mobile devices 10 shown in Figures 1 and 2 are for illustrative purposes only and various other mobile devices 10, presently known or unknown are equally applicable to the following examples.

[0035] Movement, navigation, and/or scrolling with use of a cursor/view positioning device 14 (e.g. trackball 14b or positioning wheel 14a) is beneficial given the relatively large size of visually displayed information and the compact size of display 12, and since information and messages are typically only partially presented in the limited view of display 12 at any given moment. As previously described, positioning device 14 – positioning wheel 14a and trackball 14b, are helpful cursor/view positioning mechanisms to achieve such movement. Positioning device 14, which may be referred to as a positioning wheel or scroll device 14a in one embodiment (Figure 1), specifically includes a circular disc which is rotatable about a fixed axis of housing and may be rotated by the end user's index finger or thumb. As noted above, in another embodiment (Figure 2) the trackball 14b comprises a multi-directional member that enables upward, downward and if desired, diagonal movements. The multi-directional movements afforded, in particular, by the trackball 14b and the presentation of icons and folders on display 12 provides the user with flexibility and familiarity of the layout of a traditional desktop computer interface. Also, the positioning device 14 enables movement and selection operations to be executed on the mobile device 10 using one hand. The trackball 14b in particular also enables both one-handed use and the ability to cause a cursor 18 to traverse the display 12 in more than one direction.

[0036] Figure 3 is a detailed block diagram of a preferred mobile station 32 of the present disclosure. The term "mobile station" will herein refer to the operable components of, for example, mobile device 10. Mobile station 32 is preferably a two-way communication device having at least voice and advanced data communication capabilities, including the capability to communicate with other computer systems. Depending on the functionality provided by mobile station 32, it may be referred to as a data messaging device, a two-way pager, a cellular telephone with data messaging capabilities, a wireless Internet appliance, or a data communication device (with or without telephony capabilities) – e.g. mobile device 10 shown in Figures 1 and 2. Mobile station 32 may communicate with any one of a plurality of fixed transceiver stations 30 within its geographic coverage area.

[0037] Mobile station 32 will normally incorporate a communication subsystem 34 which includes a receiver 36, a transmitter 40, and associated components such as one or more (preferably embedded or internal) antenna elements 42 and 44, local oscillators (LOs) 38, and a processing module such as a digital signal processor (DSP) 46. As will be apparent to those skilled in field of communications, particular design of communication subsystem 34 depends on the communication network in which mobile station 32 is intended to operate.

[0038] Mobile station 32 may send and receive communication signals over a network after required network registration or activation procedures have been completed. Signals received by antenna 44 through the network are input to receiver 36, which may perform such common receiver functions as signal amplification, frequency down conversion. filtering, channel selection, and like, and in example shown in Figure 3, analog-to-digital (A/D) conversion. A/D conversion of a received signal allows more complex communication functions such as demodulation and decoding to be performed in DSP 46. In a similar manner, signals to be transmitted are processed, including modulation and encoding, for example, by DSP 46. These DSP-processed signals are input to transmitter 40 for digital-to-analog (D/A) conversion, frequency up conversion, filtering, amplification and transmission over communication network via antenna 44. DSP 46 not only processes communication signals, but also provides for receiver and transmitter control. For example, the gains applied to communication signals in receiver 36 and transmitter 40 may be adaptively controlled through automatic gain control algorithms implemented in DSP 46.

[0039] Network access is associated with a subscriber or user of mobile station 32. In one embodiment, mobile station 32 uses a Subscriber Identity Module or "SIM" card 74 to be inserted in a SIM interface 76 in order to operate in the network. SIM 74 is one type of a conventional "smart card" used to identify an end user (or subscriber) of the mobile station 32 and to personalize the device, among other things. Without SIM 74, the mobile station terminal in such an embodiment is not fully operational for communication through a wireless network. By inserting SIM 74 into mobile station 32, an end user can have access to any and all of his/her subscribed services. SIM 74 generally includes a processor and memory for storing information. Since SIM 74 is coupled to a SIM interface 76, it is coupled to processor 64 through communication lines. In order to identify the subscriber, SIM 74 contains some user parameters such as an International Mobile Subscriber Identity (IMSI). An advantage of using SIM 74 is that end users are not necessarily bound by any single physical mobile station. SIM 74 may store additional user information for the mobile station as well, including datebook (or calendar) information and recent call information. It will be appreciated that mobile station 32 may also be used with any other type of network compatible mobile device 10 such as those being code division multiple access (CDMA) enabled and should not be limited to those using and/or having a SIM card 74.

[0040] Mobile station 32 is a battery-powered device so it also includes a battery interface 70 for receiving one or more rechargeable batteries 72. Such a battery 72 provides electrical power to most if not all electrical circuitry in mobile station 32, and battery interface 70 provides for a mechanical and electrical connection for it. The battery interface 70 is coupled to a regulator (not shown) which provides a regulated voltage V to all of the circuitry.

[0041] Mobile station 32 in this embodiment includes a processor 64 which controls overall operation of mobile station 32. It will be appreciated that the processor 64 may be implemented by any processing device. Communication functions, including at least data and voice communications are performed through communication subsystem 34. Processor 64 also interacts with additional device subsystems which may interface with physical components of the mobile device 10. Such addition device subsystems comprise a display 48 (the display 48 can be the display 12 including 12a and 12b of Figures 1 and 2), a flash memory 50, a random access memory (RAM) 52, auxiliary input/output subsystems 54, a serial port 56, a keyboard 58, a speaker 60, a microphone 62, a short-range communications subsystem 66, and any other device subsystems generally designated at 68. Some of the subsystems shown in Figure 3 perform communication-related functions, whereas other subsystems may provide "resident" or on-device functions. Notably, some subsystems such as keyboard 58 and display 48, for example, may be used for both communication-related functions, such as entering a text message for transmission over a communication network, and device-resident functions such as a calculator or task list. Operating system software used by processor 64 is preferably stored in a persistent store such as flash memory 50, which may alternatively be a read-only memory (ROM) or similar storage element (not shown). Those skilled in the art will appreciate that the operating system, specific device applications, or parts thereof, may be temporarily loaded into a volatile store such as RAM 52.

[0042] Processor 64, in addition to its operating system functions, preferably enables execution of software applications on mobile station 32. A predetermined set of applications which control basic device operations, including at least data and voice communication applications, as well as the inventive functionality of the present disclosure, will normally be installed on mobile station 32 during its manufacture. A preferred application that may be loaded onto mobile station 32 may be a personal information manager (PIM) application having the ability to organize and manage data items relating to user such as, but not limited to, e-mail, calendar events, voice mails, appointments, and task items. Naturally, one or more memory stores are available on mobile station 32 and SIM 74 to facilitate storage of PIM data items and other information.

[0043] The PIM application preferably has the ability to send and receive data items via the wireless network. In the present disclosure, PIM data items are seamlessly integrated, synchronized, and updated via the wireless network, with the mobile station user's corresponding data items stored and/or associated with a host computer system thereby creating a mirrored host computer on mobile station 32 with respect to such items. This is especially advantageous where the host computer system is the mobile station user's office computer system. Additional applications may also be loaded onto mobile station 32 through network, an auxiliary subsystem 54, serial port 56, short-range communications subsystem 66, or any other suitable subsystem 68, and installed by a user in RAM 52 or preferably a non-volatile store (not shown) for execution by processor 64. Such flexibility in application installation increases the functionality of mobile station 32 and may provide enhanced on-device functions, communication-related functions, or both. For example, secure communication applications may enable electronic commerce functions and other such financial transactions to be performed using mobile station 32.

[0044] In a data communication mode, a received signal such as a text message, an e-mail message, or web page download will be processed by communication subsystem 34 and input to processor 64. Processor 64 will preferably further process the signal for output to display 48 or alternatively to auxiliary I/O device 54. A user of mobile station 32 may also compose data items, such as e-mail messages, for example, using keyboard 58 in conjunction with display 48 and possibly auxiliary I/O device 54. Keyboard 58 is preferably a complete alphanumeric keyboard and/or telephone-type keypad. These composed items may be transmitted over a communication network through communication subsystem 34.

[0045] For voice communications, the overall operation of mobile station 32 is substantially similar, except that the received signals would be output to speaker 60 and signals for transmission would be generated by microphone 62. Alternative voice or audio I/O subsystems, such as a voice message recording subsystem, may also be implemented on mobile station 32. Although voice or audio signal output is preferably accomplished primarily through speaker 60, display 48 may also be used to provide an indication of the identity of a calling party, duration of a voice call, or other voice call related information, as some examples.

[0046] Serial port 56 in Figure 3 is normally implemented in a personal digital assistant (PDA)-type communication device for which synchronization with a user's desktop computer is a desirable, albeit optional, component. Serial port 56 enables a user to set preferences through an external device or software application and extends the capabilities of mobile station 32 by providing for information or software downloads to mobile station 32 other than through a wireless communication network. The alternate download path may, for example, be used to load an encryption key onto mobile station 32 through a direct and thus reliable and trusted connection to thereby provide secure device communication.

[0047] Short-range communications subsystem 66 of Figure 3 is an additional optional component which provides for communication between mobile station 32 and different systems or devices, which need not necessarily be similar devices. For example, subsystem 66 may include an infrared device and associated circuits and components, or a Bluetooth™ communication module to provide for communication with similarly enabled systems and devices. Bluetooth™ is a registered trademark of Bluetooth SIG, Inc.

[0048] As shown in Figure 4, memory 50 includes a plurality of applications 80 associated with a series of icons 102 (see Figure 5) for the processing of data. Applications 80 may be any variety of forms such as, without limitation, software, firmware, and the like. Applications 80 may include, for example, electronic mail (e-mail) 82, calendar program 84, storage and/or program for contacts 85, a multimedia player application 86, memo program 90, storage for messages 92, a search function and/or application 94 etc. An operating system (OS) 96, and in this embodiment a multimedia storage area 89 also reside in memory 50. The multimedia storage area 89 is generally a designated portion of memory 50 for storing multimedia files 120 that are used by the multimedia player application 86.

[0049] Returning to Figure 4, in addition to the multimedia player application 86, the mobile device's memory 50 can hold one or more multimedia files 120 that are stored in the multimedia storage portion 89. It will be appreciated that the multimedia files 120 may be loaded from an external source through a web browser or downloaded from a web site accessed through the communication system 30 or the video content may be streaming to the mobile device 10, and need not be stored directly on the mobile device 10.

[0050] It will also be appreciated that the multimedia file 120 may be streaming content that is provided to or otherwise obtained by the mobile device 10. Figure 10 illustrates an embodiment where the multimedia file 120 streams over a network and is received by the antenna element 42 and in turn the receiver 36. The streaming data is then processed by a Digital Signal Processor (DSP) 46 and passed to the multimedia player 898. It will be appreciated that the antenna element 42, receiver 36, DSP 46, and multimedia player 898 of the embodiment of Figure 10 form part of the device 10 and are the same as components in previous Figures having like reference numerals. Other components of the device 10 have been omitted for clarity; however, it is understood such components are included in the embodiment of Figure 10 and the details thereof will not be repeated.

[0051] Referring to Figure 11, the multimedia files 120 include video data 125 which includes a series of video frames 126. The multimedia file 120 also contains timestamp data 129a including a series of timestamps 129b. The timestamps 129b each represent a time at which an associated audio frame 128 and video frame 126 are to be played by the multimedia player 898. Typically frames are intended to be played back at regular intervals. For example successive frame timestamps of 0 millisecond (0ms), 40 ms, 80 ms, 120 ms, etc. would provide a frame rate of 25 frames per second (25 fps).

[0052] Some multimedia files 120 do not have timestamps 129a in the multimedia file 120. For example, a file 120 may include audio data 127 and video data 125 and a target frame rate. The target frame rate is the preferred frame rate for the multimedia file 120, such as for example 25 fps. The frame rate in combination with the video frames and audio frames 127 provides the information provided by a timestamp. For example, at 25 fps frames are to be played at 40 ms intervals. Thus, if the first frame is to be played at 0 ms then the fifth frame is to be played at 160 ms, and the respective timestamps of the first frame and the fifth frame would be 0 ms and 160 ms.

[0053] The multimedia file 120 can be encoded using MPEG encoding, for example MPEG-4; it will be appreciated, however, that the principles discussed herein are equally applicable to other encoding/decoding schemes. A further example of an encoding format for the multimedia file 120 is H.264. Decoding H.264 is particularly time consuming and can benefit significantly from application of the principles described herein.

[0054] In MPEG video encoding, a group of pictures is used to specify the order in which intra-frame and inter-frames are arranged, wherein the group of pictures is a stream of encoded frames in the video data stream 126. The frames 128 in MPEG encoding are of the following types: An I-frame (intra coded) corresponds to a fixed image and is independent of other picture types. Each group of pictures begins with this type of frame. A P-frame (predictive coded) contains difference information from the preceding I or P-frame. A B-frame (bidirectionally predictive coded) contains difference information from the preceding and/or following I or P-frame. D frames may also be used, which are DC direct coded pictures that serve the fast advance. In the following examples, video data stream 126 having I, B and P frames is used. It will be appreciated that the dynamic post-processing discussed below may be applied on a frame by frame basis or for every group of pictures.

[0055] Referring to Figure 9, a multimedia player 898 has an audio player 87 and a video player 88, parser 900, media clock 902, and media player client 904. The multimedia player 898 has an input for receiving multimedia file 120 as a multimedia stream 910, an output to a sound system 912, such as speaker 60 (Figure 3), an output to a display 48, and an input/output to a control interface 914, which may include for example display 48, speakers, keyboard 58 and other input /output components of the device 10.

[0056] The media player client 904 displays the media player interface 132 displayed on display 48 (Figures 6-12) and receives action requests, for example, play, seek, stop, and pause from for example from trackball 14b. The display 48 and trackball 14b are examples of a control interface 914 external to the multimedia player 898. The external control interface 914 could also provide external control through an API or other remote control mechanism. The media player client 904 passes appropriate commands to the various components of the multimedia player 898 to act in accordance with the received requests in processing multimedia file 120.

[0057] The parser 90 parses the multimedia file 120 incoming as a multimedia stream 910 into an audio stream 920 and a video stream 922. The output of the parser 90 is a compressed video stream 923a and a compressed audio stream 921aThe audio stream 920 is decoded, including decompressions, to a decoded audio stream 921b. Then the audio stream 920 is rendered to a rendered audio stream 921c. Each of the compressed audio stream 921a, the decoded audio stream 921b, and the rendered audio stream 921c form part of the audio stream 920. The video stream 922 is decoded, including decompression, to a decoded video stream 923b. Then the video stream 922 is rendered to a rendered video stream 923c. Each of the compressed video stream 923a, the decoded video stream 923b, and the rendered video stream 923c form part of the video stream 922.

[0058] Referring to Figure 12, audio stream 920 comprises audio data 127 in frames 128. Audio stream 920 also includes audio timestamps 924, one timestamp 926 for each of the frames 128. The audio timestamps 924 are derived from the multimedia timestamps 129a. Referring to Figure 13, video stream 922 comprises video data 125 in frames 125. Video stream 922 also includes video timestamps 928, one timestamp 930 for each frame 126. The video timestamps 928 are derived from the multimedia timestamps 129a.

[0059] Referring again to Figure 9, the audio player has an audio decoder 916 and audio renderer 918. The audio decoder 916 receives the audio stream 921a from the parser 900. The audio decoder 916 decodes the audio stream 920. Decoding involves decompressing the audio stream 921a. For example, an MPEG compressed audio stream 921a is decompressed from compressed format, such as MPEG, to an uncompressed audio stream 921b, such as a pulse code modulation (pcm) format. The decoded audio stream 921b is passed from the audio decoder 916 to the audio renderer 918.

[0060] The audio renderer 918 queues the audio frames 128 according to the timestamps 924. Depending on the implementation, the audio renderer 918 can perform additional rendering functions, such as digital to analog to conversion using a digital to analog converter (DAC), not shown, and the rendered data stream is an analog stream that could be played directly by speakers as sound system 912. Alternatively, where the sound system 912 has additional processing capability then the renderer 918 can queue the frames 128 and transmit the frames 128 to the sound system 912. The sound system 912 is then responsible for any digital to analog conversion. When the renderer 918 sends pcm data for a frame 128, either internally in the multimedia player 898 for further processing, or externally to the sound system 912, the audio renderer 918 updates the media clock 902 to the corresponding audio timestamp 926 for the frame 128. The audio renderer 918 can modify the audio timestamp 926 prior to sending the timestamp 926 to the media clock 902 to take into account anticipated delays introduced by the sound system 912. The media clock 902 stores the timestamp 926 as a media time 919.

[0061] Similarly, the video player 88 has a video decoder 140, sometimes referred to as a video codec, and a video renderer 940. The video decoder 140 decodes the video stream 923a. For example, an MPEG compressed video stream can be decompressed to frames in a standard NTSC TV transmission format (yuv420). The decoded video stream 921b is passed from the video decoder 140 to the video renderer 940. The video renderer 940 queues the decoded video frames 126 for transmission to the display 48. Depending on the capabilities of display 48, the frames 126 can remain in digital format, or can be converted to analog format, by the video renderer 940.

[0062] Video player 88 has a video scheduler 950. The video scheduler 950 can have access to the video stream 923b. The video scheduler 950 controls transmission of the decoded video stream 923b to the video renderer 940.

[0063] The video player 88 includes one or more video post-processing blocks 960. The video scheduler 950 commands operation of the post-processing blocks 960. The video post-processing blocks 960 can perform such functions as deblocking, color conversion, scaling, and rotation among other things. As an example, the post-processing blocks 960 in video player 88 include a deblocking filter module 144 and a color conversion module 962.

[0064] The post-processing blocks are not required to be physically or logically grouped together. For example, the deblocking filter module 144 and the video decoder 140 can be grouped together in a video decode module 980. Decoding of an H.264 video stream is typically architected to have a decode module 980 that incorporates both a video decoder 140 and a deblocking module 144. H.264 is a block oriented encoding. The strength of a deblocking filter 124, or turning off (disabling) deblocking (not using a filter 124) can be specified in the multimedia file 120, possibly in the video content itself and in the resulting video stream 922.

Deblocking can be specified by a flag encoded into the video stream. Turning off deblocking can cause the deblocking module 144 to ignore (override) the flag and skip deblocking.

[0065] The decode module 980 controls passage of the video stream from the video decoder 140 to the deblocking module 144 and to an output of the decode module 980. Similarly, the color conversion module and be grouped together with the video renderer 940 in a video render module 982. An output of the decode module 140 can be a deblocked decoded video stream 923b, which would form the input to the video render module 982. An output of the render module can be a color converted rendered video stream 923c for transmission to display 48. The render module 940 controls passage of the video stream 923c to an output of the render module 982.

[0066] Where a decode module 980 and render module 982 architecture are used the video scheduler 950 may only have access to the video stream 923b between the decode module 980 and the render module 982. The video scheduler 950 can command operation of the deblocking filter module 144 via control 984. As an example, where decode module 980 is implemented incorporating a computer program, deblocking filter module 144 can be exposed to the video scheduler through an application programming interface (API). The API can allow control of simple functionality such as deblocking on and deblocking off. Alternatively, the deblocking filter module can have a plurality of deblocking filters124 (FILTER 1 through FILTER N) and a filter selection module 142. The control can allow selection of a particular filter 124 through the filter selection module 124.

[0067] Similarly, the video scheduler 950 can command operation of the color conversion module 962 via control 986. As an example, where render module 982 is implemented incorporating a computer program, color conversion module 962 can be exposed to the video scheduler 950 through an application programming interface (API). The API can allow control of simple functionality such as color conversion on and color conversion off. It is noted that turning color conversion off is likely not a practical option in most circumstances as frames 126 will not be rendered sufficiently well for viewing. Alternatively, the color conversion module 962 can have a plurality of color conversion converters 988 (CONVERTER 1 through

CONVERTER N) and a converter selection module 990. Each converter 988 is based on a different color space conversion algorithm resulting in different quality and processing complexity. The control 986 can allow selection of a particular converter 988 through the converter selection module 990.

[0068] The video scheduler 950 has access to video timestamps 126 of video frames 126 decoded by the video decoder 140. For example, the video scheduler 950 can access decoded video timestamps 930 by accessing the video stream 923b after the video decoder 140. In the decode module 980 and render module 982 architecture described above, the video scheduler 950 can access the video stream 923b output from the decode module 980.

[0069] In alternate embodiments the video scheduler 950 could access video timestamps 930 of the decoded video stream 923b elsewhere in the video stream 923b after the video decoder 140. For example, the video scheduler 950 could access video timestamps 930 from the decoded video stream 923b after the decoder 140 and prior to the video post-processing blocks 960. As a further example, video timestamps 930 could be accessed between post-processing blocks 960, or between the post-processing blocks 960 and the video renderer 940. As another example, the video scheduler could access the video timestamps 930 directly from the video decoder 140 as represented by command line 983 between the video decoder 140 and the video scheduler 950. Direct access of the video timestamps 930 from the video decoder 140 after decoding of the video stream 922 is accessing the decoded video stream 923b.

[0070] The video scheduler 950 could access timestamps 126 from multiple locations in the video stream 923b during processing; however, there is a trade-off between overhead required to perform access in multiple location and the benefits derived thereby. It has been found that accessing the video timestamps 126 in a single location after the video decoder 140 can be sufficient for the purposes described herein.

[0071] Video post-processing improves the quality of the decoded video stream 923b from the video decoder 140 before the decoded video stream 923b reaches the video renderer 940. Video post-processing includes any step within the video player 88 post decoding (after decoding by the video decoder 140). Although video post-processing can improved the picture quality of the video frames 126 when viewed on the display 48, video post-processing can be resource intensive leading to a lag between the audio stream 920 and the video stream 922.

[0072] The video scheduler 950 has access to the media clock 902 via a request line 996. Through the media clock 902 the video scheduler 950 has access to the media time 919. The video scheduler 950 can request the media time 919 from the media clock 902. The video scheduler 950 can compare the media time to the video timestamp 930.

[0073] The video scheduler 950 has one or more thresholds 988 (THRESHOLD 1 to THRESHOLD N). The video scheduler 950 can use the thresholds 988 to trigger adaptive post-processing to increase decoded video stream 923b processing speed. The video scheduler 950 can trigger multiple post-processing adaptations based on the different thresholds. The post-processing adaptations can escalate in accordance with escalating thresholds 988.

[0074] The multimedia player 898 can be implemented in a variety of architectures. As described herein, the multimedia player 898 is a combination of the media player application 86 running on processor 64 to carry out the functions described herein. The processor 64 can be a general purpose processor without dedicated hardware functionality for multimedia processing. Such a general purpose processor becomes a special purpose processor when executing in accordance with the multimedia player application 86. Alternatively, any components of the multimedia player 898 can be implemented entirely or partially within dedicated hardware. For example, audio renderer 918 can comprise a DAC as described elsewhere herein. Asa afurther example, such hardware can be incorporated into a single integrated circuit, or distributed among a plurality of discrete integrated circuits. Each of these hardware configurations is understood to be a processor 64, whether or not the functionality is distributed among various hardware components.

[0075] Referring to Figure 14, the video scheduler 950 can access the video timestamp 930 of decoded video stream 923b, and check the media time 919 to determine the amount of time the timestamp 930 of the decoded video stream 923b is behind the media time 919 (the "video lag"). In the example shown in Figure 14, nine audio frames 128 have been rendered, and the audio timestamp 926 of the rendered audio stream 921c is nine. The media time 919 is correspondingly nine. The video decoder 140 has decoded eight video frames 126. Accordingly, the video timestamp 930 is eight. The video lag 996 is therefore one (nine minus 8).

[0076] Assuming the target frame rate is 40 ms then the video decoding is lagging the audio rendering by 40 ms. As the video timestamp 930 is for decoded video and the rendered audio timestamp 926 is for rendered audio, a lag of one frame 126 should be acceptable as there should be a rendered video frame 126 ahead of the decoded video frame 126, and the video stream 922 is synchronized with the audio stream 920. Accordingly, a lag of one frame (or 40 ms in the example) between the media time 919 and the decoded video timestamp 930 provides a first threshold for determining unacceptable lag.

[0077] Referring to Figures 15-18, additional lag examples of two frames 126 (Figure 15), four frames 126 (Figure 16), six frames 126 (Figure 17), and eight frames 126 (Figure 18). The reference numerals from Figure 14 apply equally to Figures 15-18 and are not repeated in Figures 15-18.

[0078] Referring to Figures 14-18, if first, second, third and fourth thresholds of greater than one frame, greater than three frames, greater than five frames and greater than seven frames are used then the example of Figure 14 will be less than the first threshold, the example of Figure 15 will be greater than the first threshold and less than the second threshold, the example of Figure 16 will be greater than the second threshold and less than the third threshold, the example of Figure 17 will be greater than the third threshold and less than the fourth threshold, and the example of Figure 18 will be greater than the fourth threshold. Each of the examples of Figures 15-18 are greater than the first threshold. Using the 40 ms per frame rate discussed earlier, the first threshold could be 70 ms, the second threshold 150 ms, the third threshold 220 ms, and the fourth threshold 300 ms. Different numbers of thresholds, different threshold values for each threshold, and different actions to be taken to increase video stream processed speed could be used as desired, taking into account target frame rates, acceptable lag, point in the decoded video stream 923b where the video timestamp 930 is accessed, and the affect that changes in post-processing will have on processing speed of the video stream 922.

[0079] Escalating thresholds, examples of which are discussed above, can be used to assess the degree of video lag 996. Escalating action can be taken as higher thresholds are reached.

[0080] Referring to Figure 19, a method of operating multimedia player 898 is shown in flowchart form. The method can be embodied as instructions in one or more computer programs, such as multimedia player application 86, executing on a computer, such as mobile device 10. At 800, parser 900 parses multimedia file 120 into audio stream 920 and video stream 923a. At 802, audio stream 920 is decoded. At 804, decoded audio stream 920 is rendered. At 806, the media time 919 is updated with an audio timestamp 926 of the rendered audio stream 920 as the audio stream 920 is rendered. At 810, while the audio stream 920 is decoded and rendered, the video stream is decoded. Thus, the video stream 922 is being processed in parallel with the audio stream 920. At 812, a timestamp 930 of the decoded video stream 923b is compared against the media time 919. It is not necessary to compare the timestamp 930 on a video frame 126 by video frame 126 basis. As an alternative, the timestamp 930 can be compared less often, for example at 1000 ms intervals to reduce processing load on the mobile device 10. Longer intervals allow for adaptive post-processing to have an opportunity to take effect before comparing again to take escalating action if necessary. The adaptive post-processing settings will apply once set until reset at a subsequent interval.

[0081] At 814, if the video timestamp 930 is within a first threshold of the media time 919 then, at 816, default post-processing is allowed to continue. Default post-processing could include all post-processing blocks with best quality. For example, for decoding of H.264 encoding default processing could include enabling (turning on) debloding. Again at 814, if the video timestamp 930 is above a first threshold of the media time 919 then, at 818, default post-processing is adapted to increase video stream 922 processing speed. At 820, following post-processing at 816 or at 818 the video stream 923b is rendered. As will later be described herein, adapting post-processing can involved skipping, or discarding, a frame 126 in the video stream 923b so that the frame 126 is not post-processed at all and is not rendered, or is partially post-processed and not rendered. An example of a circumstance where a frame 126 could be partially processed, but not rendered could include decoding an H. 264 encoded frame and deblocking the frame, and then discarding the frame 126.b Preferably a frame 126 that is to be discarded would be discarded at an early a stage after decoding as possible; although, accessing frames 126 prior to some post-processing may not be practical in all circumstances.

[0082] Referring to Figure 20, adapting post-processing at 818 of Figure 19 can include, at 850, discarding frames. For example, video scheduler 950 can access the decoded video stream 923b to prevent video frames 126 from being rendered. Discarding frames can reduce overall processor load and increase video stream processing speed. Also, discarding frames can allow a backlog in later post-processing or rendering to clear so that decoded video frames 126 can be released from the video decoder 140.

[0083] As adaptive post-processing including discarding frames 126 will apply from one checking of the media time 919 to the next, it will likely not be desirable to discard all frames. Rendered video could be blank or jittery, and content could be lost. As an alternative to discarding all frames 126, frames 126 could be regularly discard. For example, every third frame could be discarded. If a chosen period for discarding frames 126 is insufficient to synchronize rendering of the video stream 922 and audio stream 920 then the period could be decrease (making the period shorter), for example from every third frame 126 to every second frame 126. Discarding frames periodically reduces the actual frame rate. For example if the target frame rate for a multimedia file is 30 fps then discarding every third frame 126 will result in an actual frame rate of 20 fps, and discarding every second frame will result in an actuaol frame rate of 15 fps. Depending on the actual video lag, the percentage of discarded frames can be increased to a maximum of 100%.

[0084] Referring to Figure 21, adapting post-processing at 818 of Figure 19 can include, at 852, degrading quality of color space conversion. Color space conversion can be degraded by selecting a color space converter 988 of lower quality and faster processing time. Again, overall processing load can be decreased. Processing time for color space conversion can be reduced to clear video processing backlogs.

[0085] Referring to Figure 22, adapting post-processing at 818 of Figure 19 can include, at 854, degrading deblocking. Degrading deblocking can include selecting a deblocking filter 124 of lower quality and faster processing time. Alternatively, degrading deblocking can include turning deblocking off. For example when decoding H. 264 encoding an otherwise mandatory deblocking filter 124 is disabled even if a video stream 922 signals that H.264 deblocking is mandatory for the stream.

[0086] Referring to Figure 23, an escalating method of adapting post-processing of a decoded video stream 923b which may be incorporated in multimedia player 898 includes combining a plurality of escalating thresholds for decoded video lag from rendered audio and, for each threshold, differently adapting post-processing to increase video processing speed. For example, adapting post-processing at 818 of Figure 19 can include, at 880, if decoded video timestamp 930 is greater than a first threshold and less than a second threshold from media time 919 then, at 882, degrade color space conversion quality. At 880, if video timestamp 930 is not within a second threshold of media time 919, and, at 884, if video timestamp 930 is within a third threshold of media time then, at 886, video frames 126 in the decoded video stream 923b can be discarded. At 884, if video timestamp 930 is not within a third threshold of media time 919, and, at 888, if video timestamp 930 is within a fourth threshold of media time then, at 890, video frames 126 in the decoded video stream 923b can be discarded as discussed previously and deblocking degraded for those frames 126 that are not discarded. At 888, if video timestamp 930 is not within a fourth threshold of media time 919, then, at 892, deblocking can be degraded, at 894 video frames 126 can be discarded, and at 896 quality of color space conversion can be degraded.

[0087] It is noted that the methods described herein can be combined with other triggers for increasing video processing speed, such as processor 64 load. For example, video scheduler 950 can be connected to processor 64 as indicated by command line 998 or some other component of the device 10 that provides an indication of processor 64 usage by the multimedia player 898 or components thereof. Depending on the level of processor 64 usage, the video scheduler 950 can apply adaptive post-processing including multiple thresholds 999 based on processor 64 usage to trigger escalating post-processing adaptations, examples of which have been described herein. Example structures and methods to access processor load are disclosed in US Pat. Pub. No. 2009/0135918 of Mak-Fan et al having a Pub. Date of May 28, 2009, and in US Pat. Pub. No. 2009/0052555 of Mak-Fan et al having a Pub. Date of Feb. 26, 2009.

[0088] The video scheduler 950 can use the thresholds 988 to trigger adaptive post-processing to increase decoded video stream 923b processing speed. The video scheduler 950 can trigger multiple post-processing adaptations based on the different thresholds. The post-processing adaptations can escalate in accordance with escalating thresholds 999. Specific numbers of thresholds and values for those thresholds to be triggered based on processor 64 usage will depend on the performance desired.

[0089] Referring to Figure 24, an escalating method of adapting post-processing of a decoded video stream 923b based on processor 64 load may be incorporated in multimedia player 898 including combining a plurality of escalating thresholds for decoded video lag from rendered audio and, for each threshold, differently adapting post-processing to increase video processing speed. The method is similar to that of Figure 19. The discussion of like steps having like reference numerals will not be repeated. Steps 812 and 814 of Figure 19 are replaced by steps 1000 and 1002 respectively. At step 1000, instead of checking the media clock 902, the processor 64 load is checked. At step 1002, the threshold 999 is a processor load threshold 999 and the processor load is checked against the threshold 999. The method of Figure 24 continues with the other steps of Figures 20-22 for single threshold, single adaptation examples. With regard to Figure 23 and multiple threshold to multiple post-processing adaptations, in steps 880, 884, and 888 the processor 64 load is compared against a processor 64 load threshold 999 in place of comparing the relationship between the video timestamp 930 and the media time 918 to the threshold 999.

[0090] It is noted that the multiple adaptations in the example shown in Figure 23 as it applied to video lag triggered adaptive post-processing discussed above with reference to Figure 19 and to processor load triggered adaptive post-processing as discussed above with reference to Figure 24, include post-processing adaptations of different post-processing steps. The different post-processing steps include, for example, transferring of decoded frames 126 for rendering (at which time frame discarding can be applied), deblocking (at which time deblocking can be degraded), converting color space (at which time color space conversion quality can be degraded).

[0091] Turning now to Figure 5, the mobile device 10 displays a home screen 100, which is preferably the active screen when the mobile device 10 is powered up and constitutes the main ribbon application. The home screen 100 generally comprises a status region 104 and a theme background 106, which provides a graphical background for the display 12. The theme background 106 displays a series of icons 102 in a predefined arrangement on a graphical background.

[0092] In some themes, the home screen 100 may limit the number icons 102 shown on the home screen 100 so as to not detract from the theme background 106, particularly where the background 106 is chosen for aesthetic reasons. The theme background 106 shown in Figure 5 provides a grid of icons. In other themes (not shown), a limited list of icons may be displayed in a column (or row) on the home screen along one portion of the display 12. In yet another theme, the entire list of icons may be listed in a continuous row along one side of the home screen on the display 12 enabling the user to scroll through the list while maintaining a limited number of currently visible icons on the display 12. In yet another theme (not shown), metadata may be displayed with each of a limited number of icons shown on the home screen. For example, the next two appointments in the user's calendar may be accessed by the processor 64 and displayed next to the calendar icon. It will be appreciated that preferably several themes are available for the user to select and that any applicable arrangement may be used.

[0093] One or more of the series of icons 102 is typically a folder 112 that itself is capable of organizing any number of applications therewithin.

[0094] The status region 104 in this embodiment comprises a date/time display 107. The theme background 106, in addition to a graphical background and the series of icons 102, also comprises a status bar 110. The status bar 110 provides information to the user based on the location of the selection cursor 18, e.g. by displaying a name for the icon 102 that is currently highlighted.

[0095] Accordingly, an application, such as the media player application 88 may be initiated (opened or viewed) from display 12 by highlighting a media player icon 114 using the positioning device 14 and providing a suitable user input to the mobile device 10. For example, media player application 88 may be initiated by moving the positioning device 14 such that the contacts icon 114 is highlighted as shown in Figure 5, and providing a selection input, e.g. by pressing the trackball 14b.

[0096] Turning now to Figure 6, a media player interface 132 is shown, which is displayed on display 12 upon initiating the media player application 88. Within the interface 132, decoded video content is displayed. Figure 6 shows the general arrangement of pixels 134 that define the content of a video frame 128. A blocking artefact 136 is also shown by way of example, which, as can be seen, is substantially larger than an individual pixel 134 in this example and, as explained above, may cause a distraction for the user as they watch the video. Figure 6 generally shows a condition where either a very low-complexity deblocking filter 124 is used, or where a deblocking filter 124 is not used or is 'turned off'.

[0097] Figure 7 shows the same media player interface 132, a reduced blocking artefact 136' that uses a slightly more complex deblocking filter 124 when compared to that shown in Figure 6.

[0098] Figure 8 shows yet another reduced blocking artefact 136'', which is closer to the actual pixel size and thus does not appear as distracting as it would appear more or less similar to a regular pixel. This may be a result of having been processed by a more complex deblocking filter 124 than the previous filters or because the artefact itself is not as bad at that time. For the purposes of illustrating the principles below, it is assumed that the artefact 136 shown in Figures 6-8 is the same artefact processed according to three different blocking filters 124 having differing levels of complexity and being applied according to different system environments at the time of performing the deblocking filtering.

[0099] Various functions of the multimedia player 898 are shown in the Figures and described herein with reference to distinct components of the multimedia player 898. It is recognized that the functions of the components herein can be combined with other components in the media player, and the particular layout of components in the Figures are provided by way of example only.

[00100] An aspect of an embodiment of this description provides a method of operating a multimedia player. The method includes decoding an audio stream of the multimedia player and rendering the decoded audio stream in the multimedia player, updating the media time of the media player with an audio timestamp of the rendered audio stream as the audio stream is rendered, and, while decoding and rendering the audio stream, decoding a video stream and checking the media clock to determine if a video timestamp of the decoded video stream is within a threshold of the media clock time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.

[00101] The audio stream can be decoded and rendered frame by frame from the audio stream, the media clock can be updated as each frame is rendered from the audio stream, the video stream can be decoded frame by frame, and the media clock can be checked for each decoded video frame.

[00102] Adapting post-processing of the video stream to decrease video stream post-processing time can include degrading color space conversion processing. Adapting post-processing of the video stream to decrease video stream post-processing time can include frame discarding. Adapting post-processing of the video stream to decrease video stream post-processing time can include degrading deblock filtering. Degrading deblock filtering can include turning deblocking off.

[00103] Prior to decoding the audio stream and decoding the video stream, the audio stream and the video stream can be separated from a multimedia stream by a parser. The video timestamp of the video stream and the audio timestamp of the audio stream can be provided as a multimedia timestamp of the multimedia stream. The video timestamp of the video stream and the audio timestamp of the audio stream can be generated by the parser when separating the audio stream and the video stream from the multimedia stream.

[00104] Another aspect of an embodiment of this description provides a computer readable medium, such as memory , can include computer executable instructions, such as media player configured for causing a mobile device, such as mobile device, to perform the methods described herein.

[00105] A further aspect of an embodiment of this description provides a multimedia player for use to render a multimedia file including audio and video content, the multimedia player including an audio decoder and audio renderer configured to output an audio timestamp and a rendered audio stream from the multimedia file, a video decoder to output a decoded video stream comprising a video timestamp, a video renderer configured to output a rendered video stream from the multimedia file, one or more post-processing blocks, a media clock configured to receive the audio timestamp from the audio renderer and store the received audio timestamp as a media time, and a video scheduler configured to check the media clock for the media time to determine if a video timestamp of the decoded video stream is within a threshold of the media time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.

[00106] Another further of an embodiment of this description provides a mobile device including a processor and a multimedia player application stored on a computer readable medium accessible to the processor, the multimedia player application comprising instructions to cause the processor to decode an audio stream of the media player and render the decoded audio stream in the media player, update a media time of the media player with an audio timestamp of the rendered audio stream as the audio stream is rendered, and while decoding and rendering the audio stream, decode a video stream and check the media time to determine if a video timestamp of the decoded video stream is within a threshold of the media time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.

[00107] In yet a further aspect of an embodiment of this description a method of operating a media player executing on a mobile device includes decoding a video stream and checking if processing of the video stream is within a plurality of thresholds, and if processing of the video stream is not within one of a plurality of thresholds then adapting post-processing of the video stream to decrease video stream post-processing, such adapting in different post-processing steps depending on the threshold exceeded by the processing of the video stream.

[00108] The plurality of threshold can include one or more thresholds based on video lag between the video stream and the audio stream within the media player. The plurality of thresholds can include one or more thresholds based on processor usage by processing of the video stream.

[00109] Other aspects and embodiments of those aspects, and further details of the above aspects and embodiments, will be evident from the detailed description herein.

[00110] Application of one or more of the above-described techniques may provide one or more advantages. For example, a user of a media player may experience a more pleasant rendition of audio and video output. Audio and video data may seem to be more in synchronization. Further, a media player may be better able to support multimedia files of a range of complexities.

[00111] It will be appreciated that the particular options, outcomes, applications, screen shots, and functional modules shown in the figures and described above are for illustrative purposes only and many other variations can be used according to the principles described.

[00112] Although the above has been described with reference to certain specific embodiments, various modifications thereof will be apparent to those skilled in the art as outlined in the appended claims.

What is claimed is:

1. A method of operating a media player executing on a mobile device, the method comprising:
    i. decoding an audio stream of the media player and rendering the decoded audio stream in the media player,
    ii. updating a media time of the media player with an audio timestamp of the rendered audio stream as the audio stream is rendered, and
    iii. while decoding and rendering the audio stream, decoding a video stream and checking the media time to determine if a video timestamp of the decoded video stream is within a threshold of the media time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.
2. The method of claim 1 wherein the audio stream is decoded and rendered frame by frame from the audio stream, the media clock is updated as each frame is rendered from the audio stream, the video stream is decoded frame by frame, and the media clock is checked for each decoded video frame.
3. The method of claim 1 or 2 wherein adapting post-processing of the video stream to decrease video stream post-processing time comprises degrading color space conversion processing.
4. The method of any one of claim 1 to 3 wherein adapting post-processing of the video stream to decrease video stream post-processing time comprises frame discarding.
5. The method of any one of claim 1 to 4 wherein adapting post-processing of the video stream to decrease video stream post-processing time comprises degrading deblock filtering.
6. The method of claim 5 wherein degrading deblock filtering comprises turning deblocking off.
7. The method of any one of claim 1 to 6 wherein, prior to decoding the audio stream and decoding the video stream, the audio stream and the video stream are separated from a multimedia stream by a parser.
8. The method of claim 7 wherein the video timestamp of the video stream and the audio timestamp of the audio stream are provided as a multimedia timestamp of the multimedia stream.

9. The method of claim 7 wherein the video timestamp of the video stream and the audio timestamp of the audio stream are generated by the parser when separating the audio stream and the video stream from the multimedia stream.
10. A computer readable medium comprising computer executable instructions configured to cause a mobile device to perform the method of any one of claim 1 to 10.
11. A multimedia player for use to render a multimedia file including audio and video content, the multimedia player comprising:
    i. an audio decoder and audio renderer configured to output an audio timestamp and a rendered audio stream from the multimedia file,
    ii. a video decoder to output a decoded video stream comprising a video timestamp,
    iii. a video renderer configured to output a rendered video stream from the multimedia file,
    iv. one or more post-processing blocks,
    v. a media clock configured to receive the audio timestamp from the audio renderer and store the received audio timestamp as a media time, and
    vi. a video scheduler configured to check the media clock for the media time to determine if a video timestamp of the decoded video stream is within a threshold of the media time, and if not then adapting post-processing of the video stream to decrease video stream post-processing time.

12. A mobile device comprising:
    i. a processor and a multimedia player application stored on a comuter readable medium accessible to the processor, the multimedia player application comprising instructions to cause the processor to:
    ii. decode an audio stream of the media player and render the decoded audio stream in the media player,
    iii. update a media time of the media player with an audio timestamp of the rendered audio stream as the audio stream is rendered, and

iv. while decoding and rendering the audio stream, decode a video stream and check the media time to determine if a video timestamp of the decoded video stream is within a threshold of the media time, and if not then adapt post-processing of the video stream to decrease video stream post-processing time.

13. A method of operating a media player executing on a mobile device, the method comprising:

i. decoding a video stream and checking if processing of the video stream is within a plurality of thresholds, and if processing of the video stream is not within one of a plurality of thresholds then adapting post-processing of the video stream to decrease video stream post-processing, such adapting in different post-processing steps depending on the threshold exceeded by the processing of the video stream.

14. The method of claim 1 wherein the plurality of threshold includes one or more thresholds based on video lag between the video stream and the audio stream within the media player.
15. The method of claim 1 wherein the plurality of thresholds includes one or more thresholds based on processor usage by processing of the video stream.

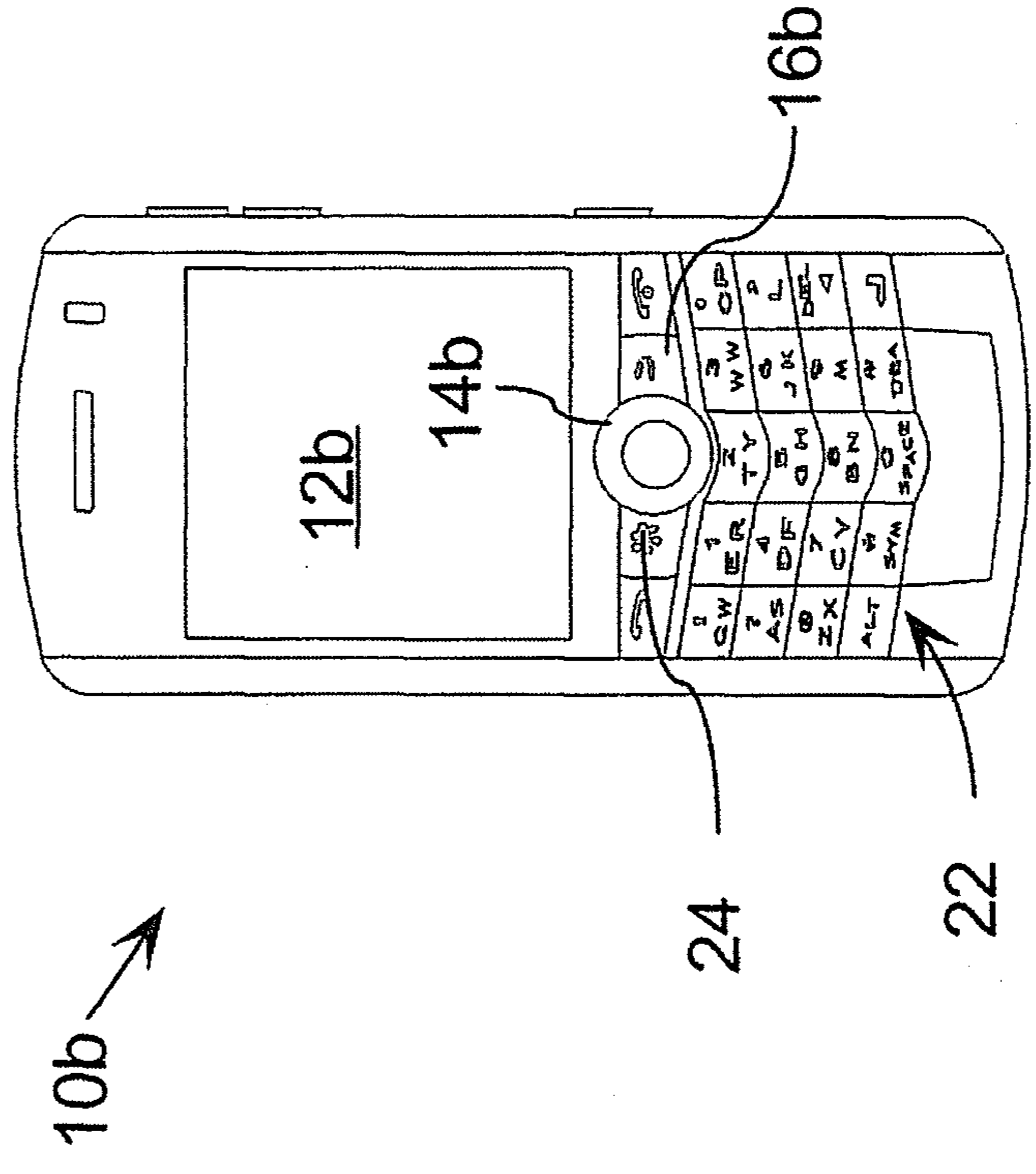


Figure 2

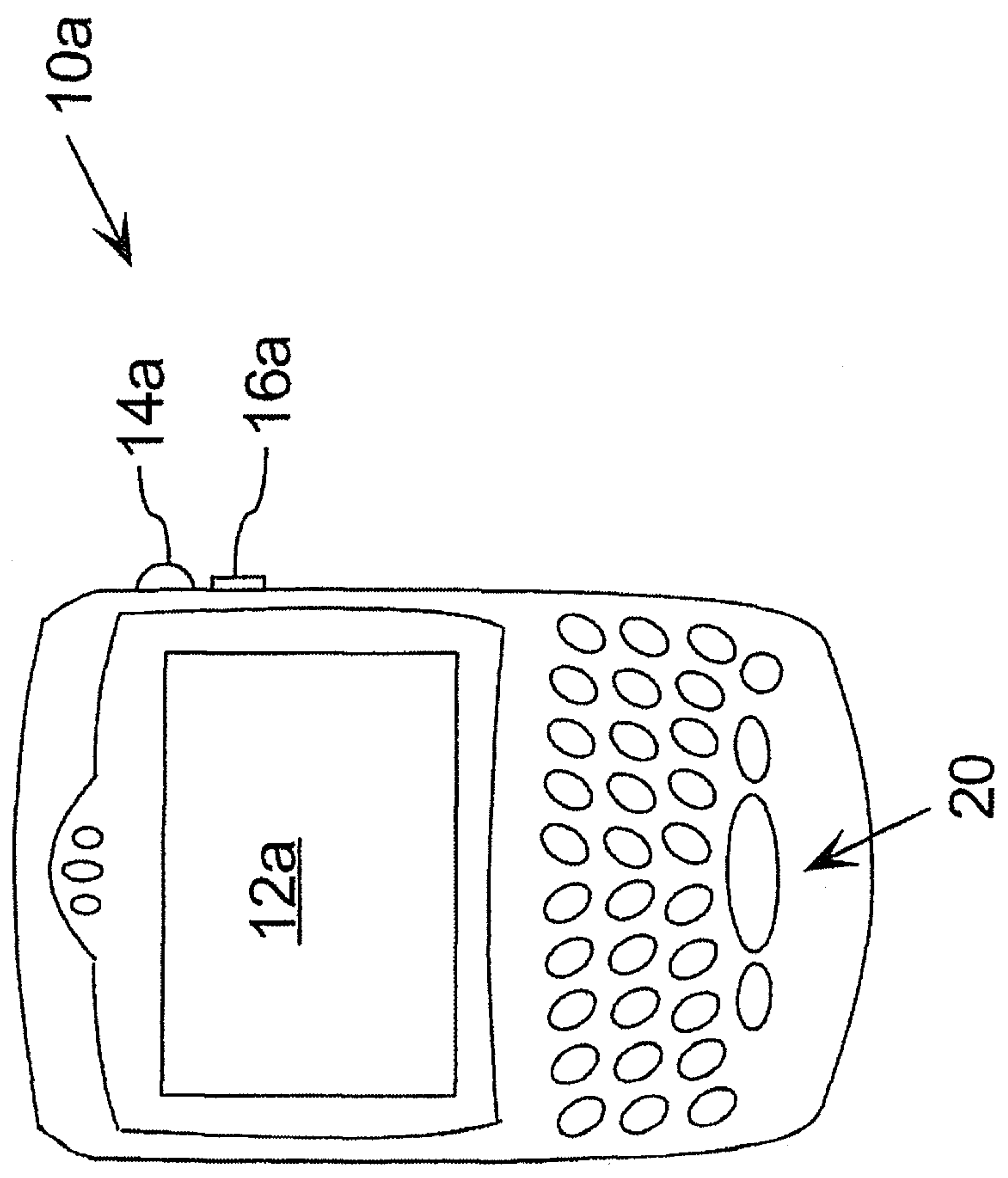
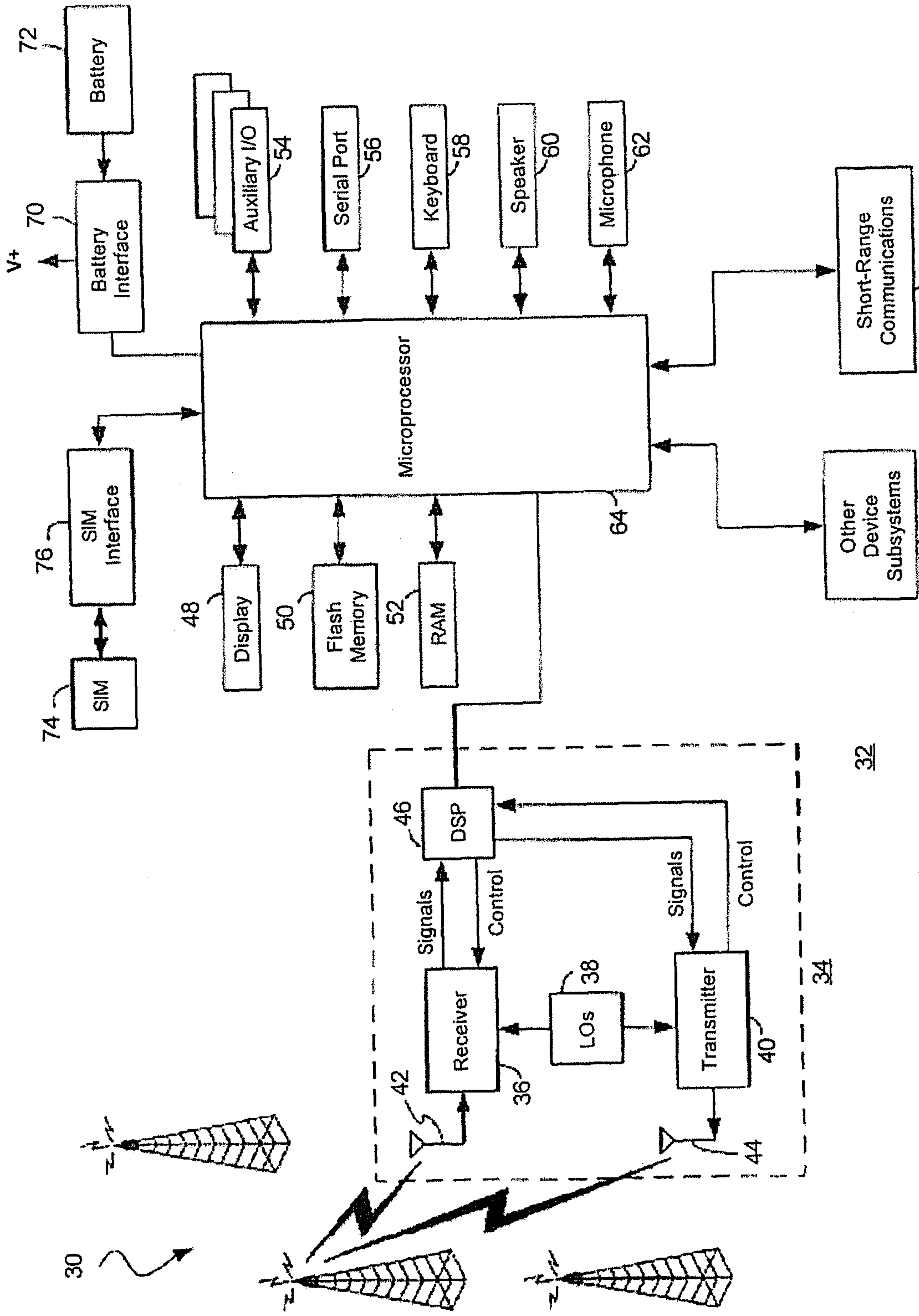
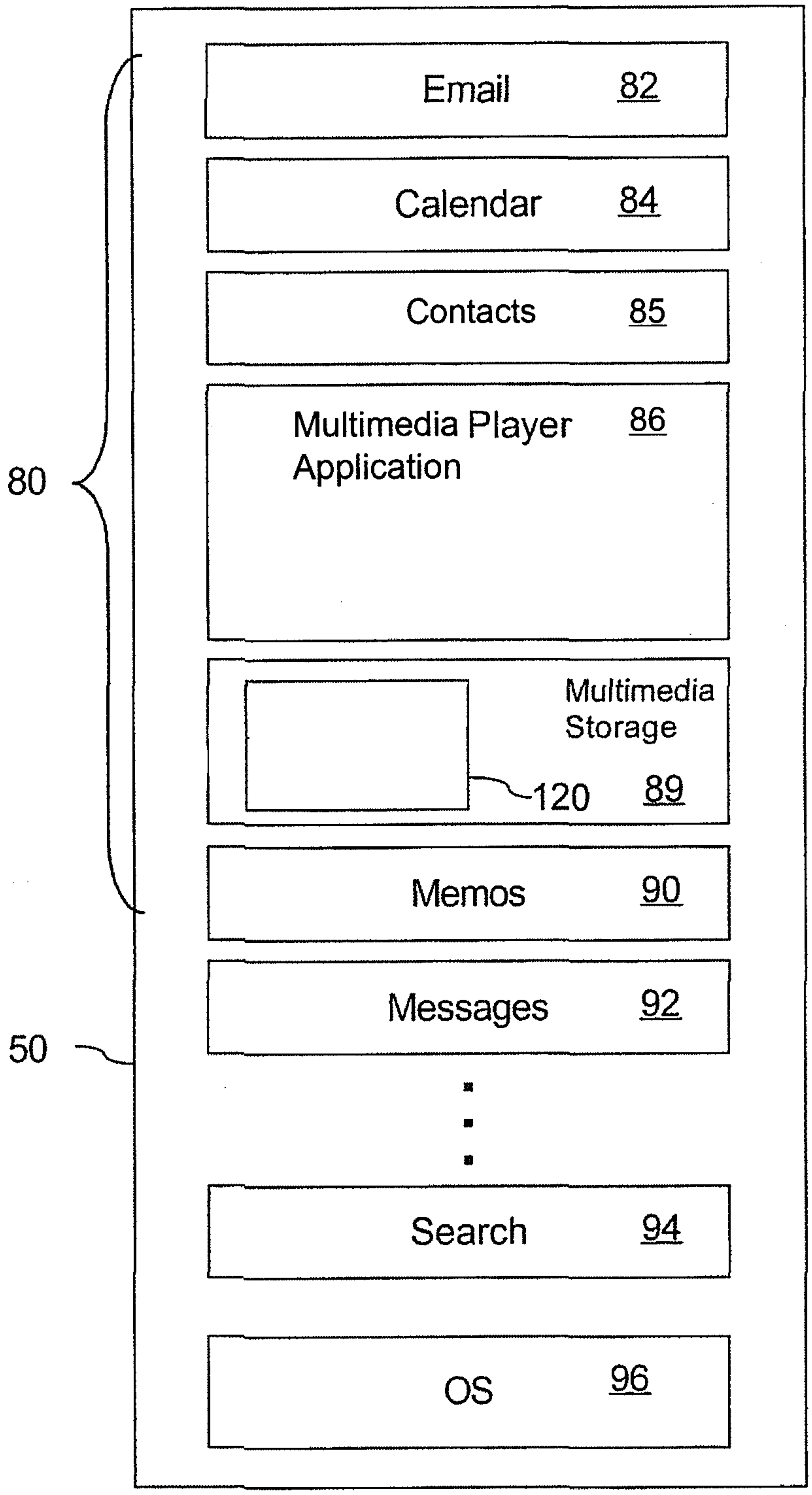


Figure 1



72
Battery
70
Battery Interface
V+
Auxiliary I/O
54
Serial Port
56
Keyboard
58
Speaker
60
Microphone
62
Microprocessor
64
Short-Range Communications
66
Other Device Subsystems
68
76
SIM Interface
74
SIM
48
Display
50
Flash Memory
52
RAM
46
DSP
Signals
Control
Signals
Control
42
Receiver
36
38
LOs
Transmitter
40
44
34
32
30

Figure 3



Email 82
Calendar 84
Contacts 85
Multimedia Player Application 86
Multimedia Storage 89
120
Memos 90
Messages 92
Search 94
OS 96
80
50

Figure 4

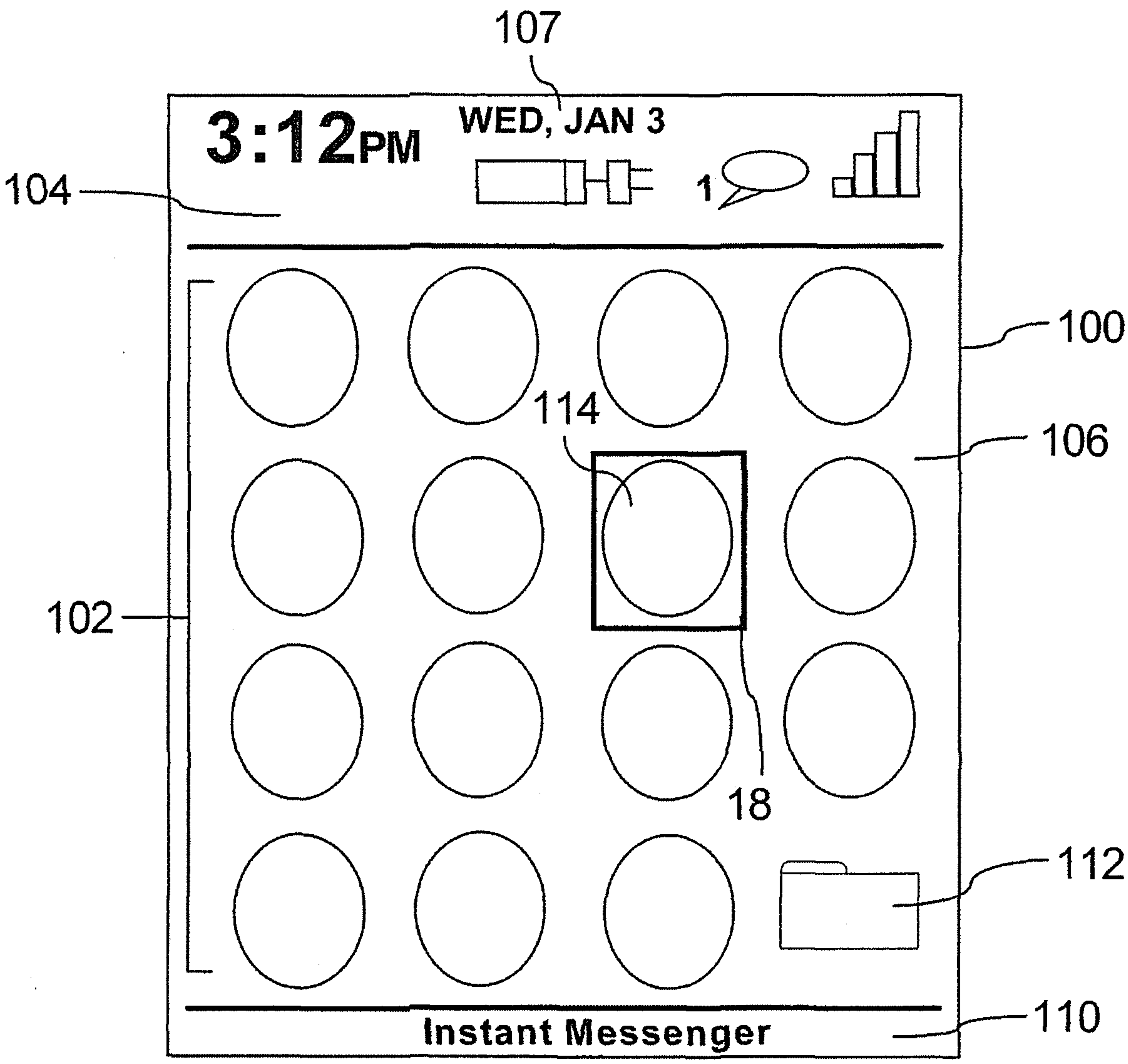

107
3:12PM
WED, JAN 3
1
104
100
114
106
102
18
112
Instant Messenger
110

Figure 5

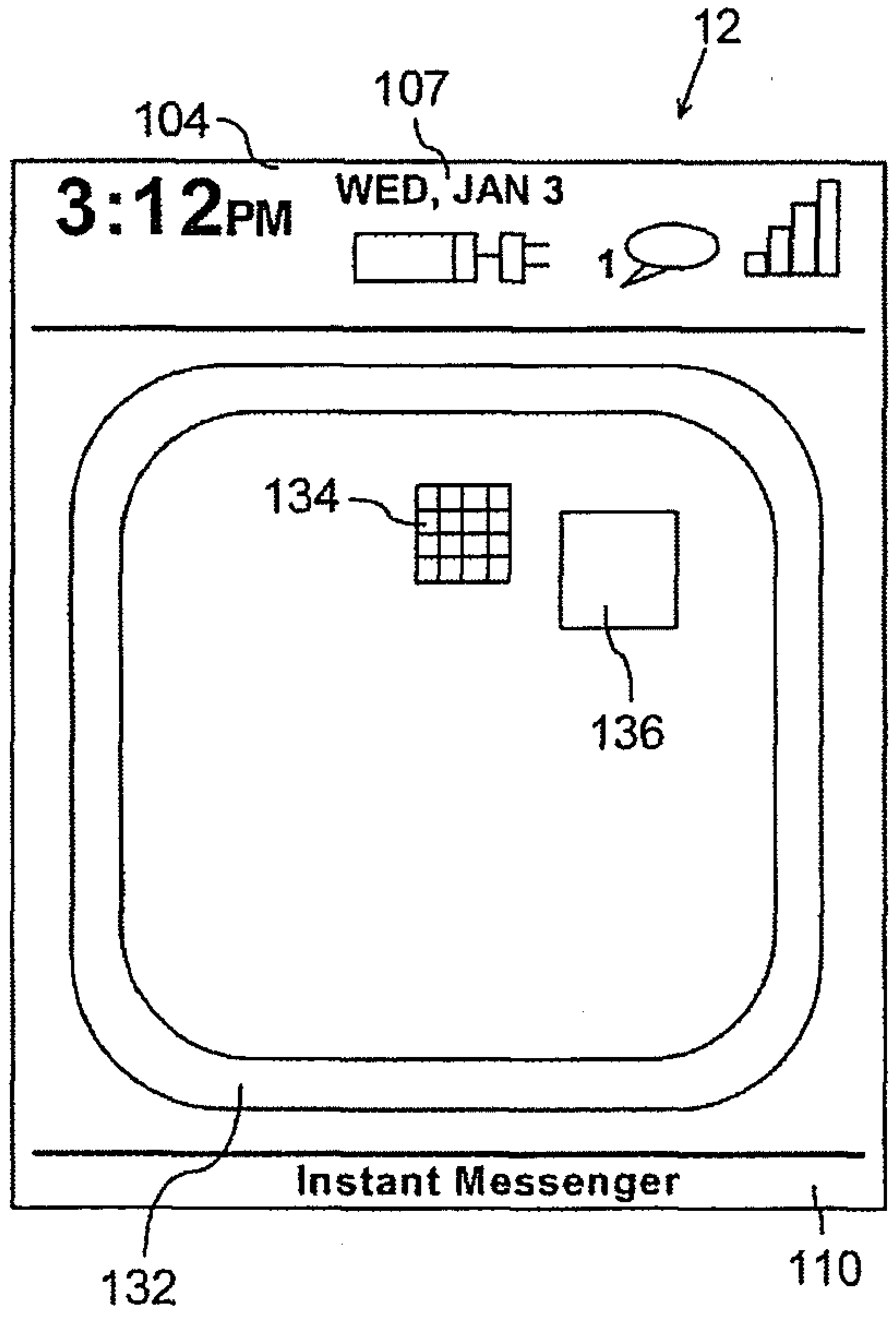


Figure 6

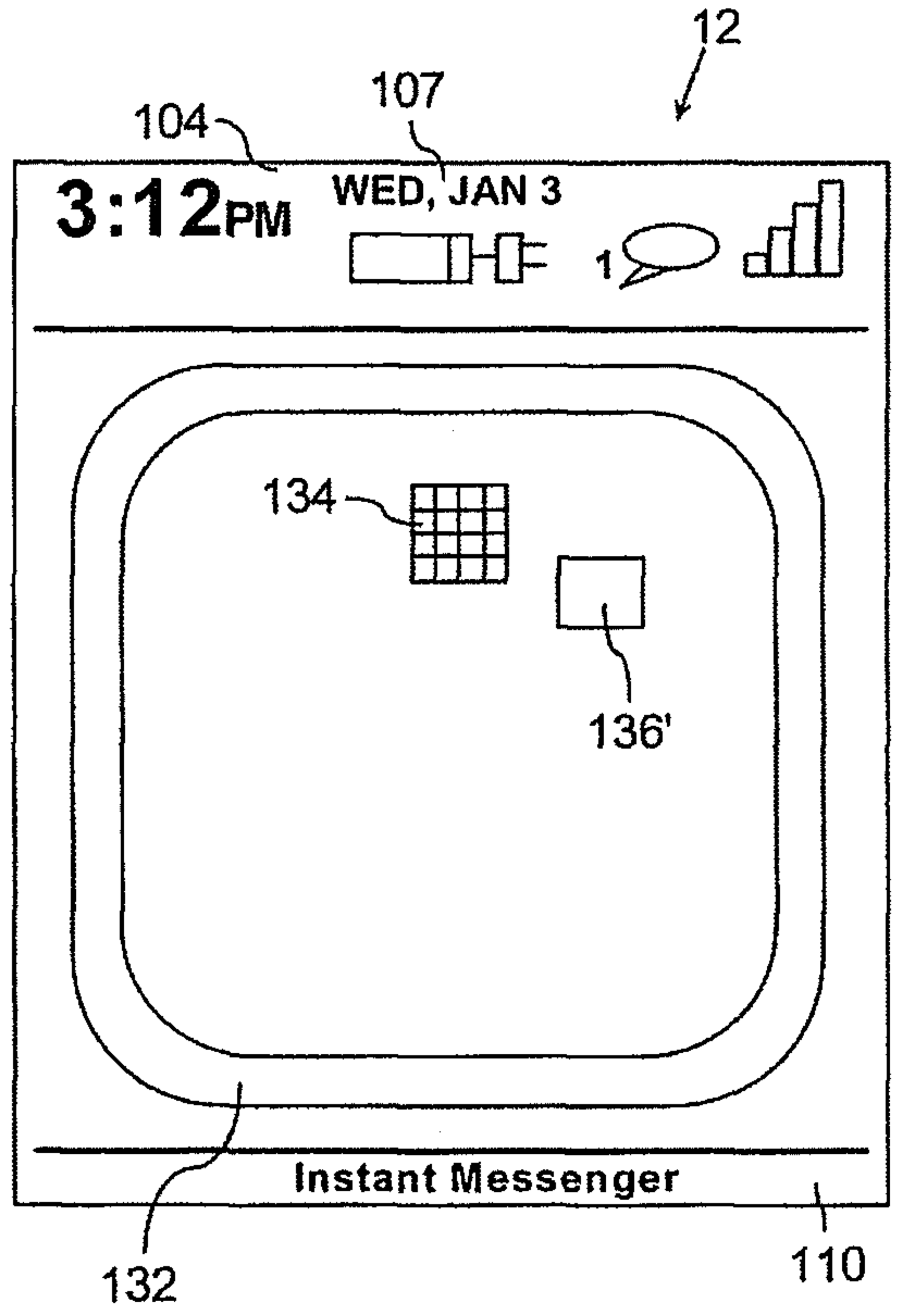


Figure 7

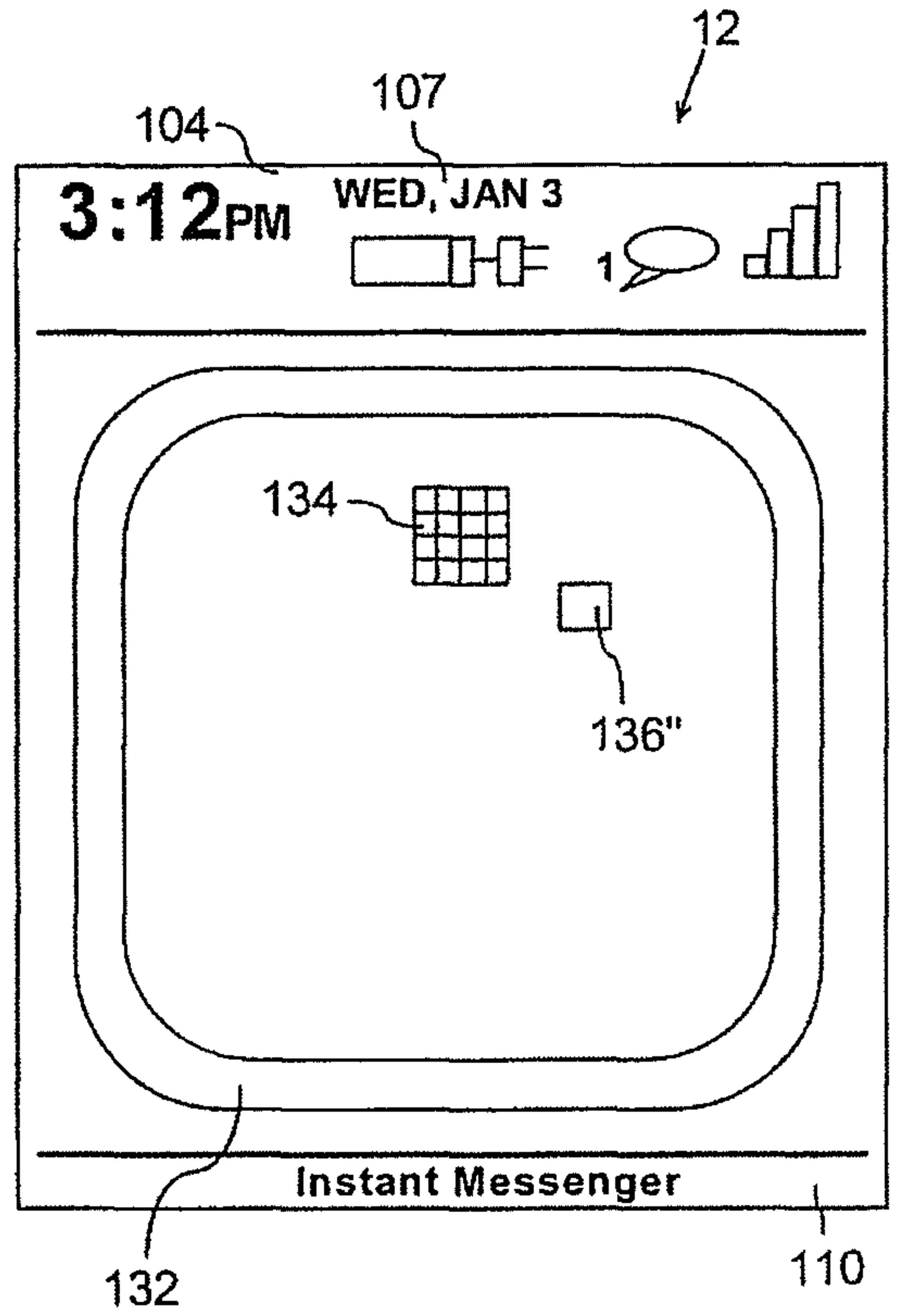


Figure 8

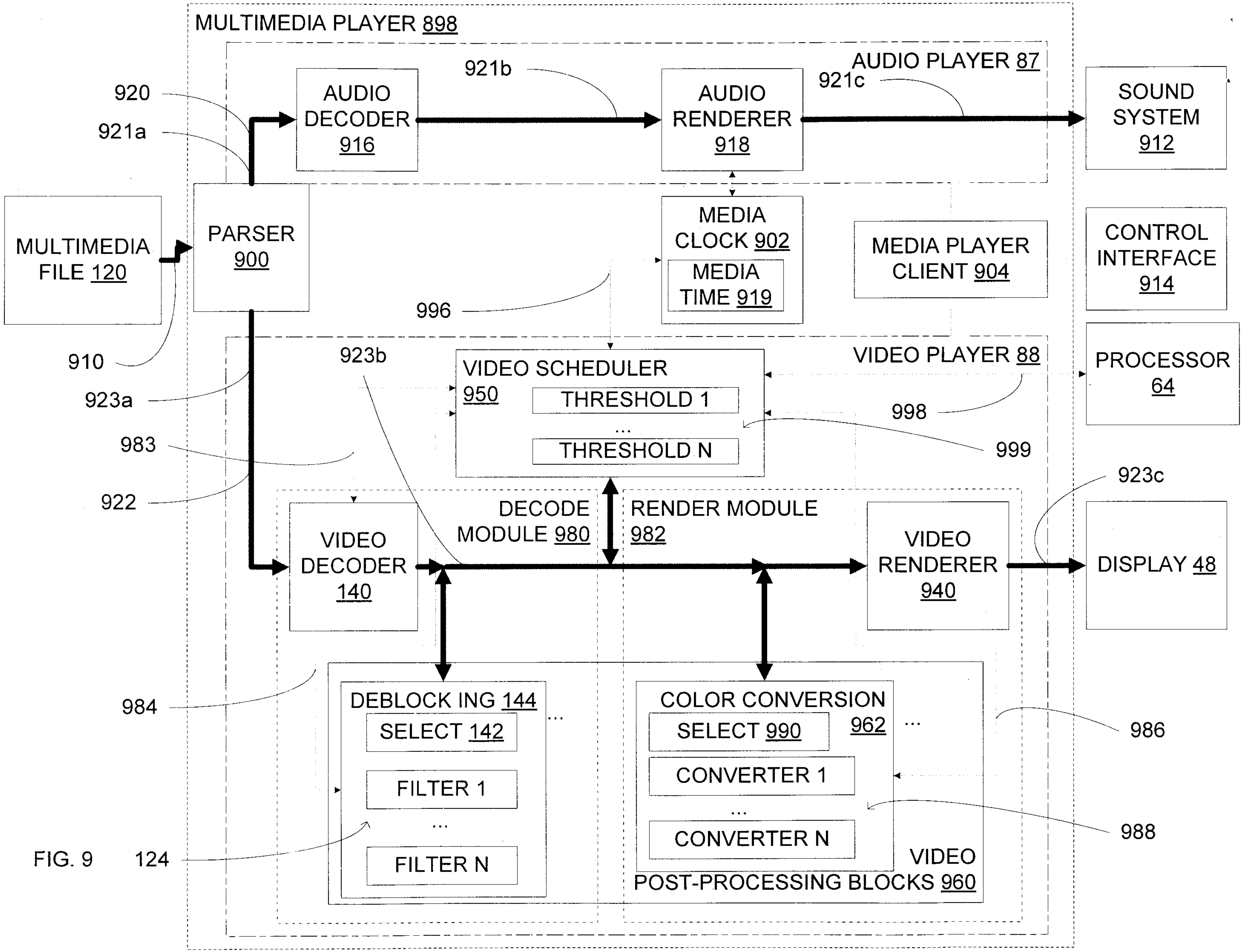

MULTIMEDIA PLAYER 898
AUDIO PLAYER 87
AUDIO DECODER 916
AUDIO RENDERER 918
SOUND SYSTEM 912
921b
921c
920
921a
MULTIMEDIA FILE 120
PARSER 900
910
MEDIA CLOCK 902
MEDIA TIME 919
MEDIA PLAYER CLIENT 904
CONTROL INTERFACE 914
996
VIDEO PLAYER 88
PROCESSOR 64
VIDEO SCHEDULER 950
THRESHOLD 1
...
THRESHOLD N
923a
923b
998
999
983
922
923c
DECODE MODULE 980
RENDER MODULE 982
VIDEO DECODER 140
VIDEO RENDERER 940
DISPLAY 48
984
DEBLOCK ING 144
SELECT 142
FILTER 1
...
FILTER N
COLOR CONVERSION 962
SELECT 990
CONVERTER 1
...
CONVERTER N
VIDEO POST-PROCESSING BLOCKS 960
986
988
124

FIG. 9

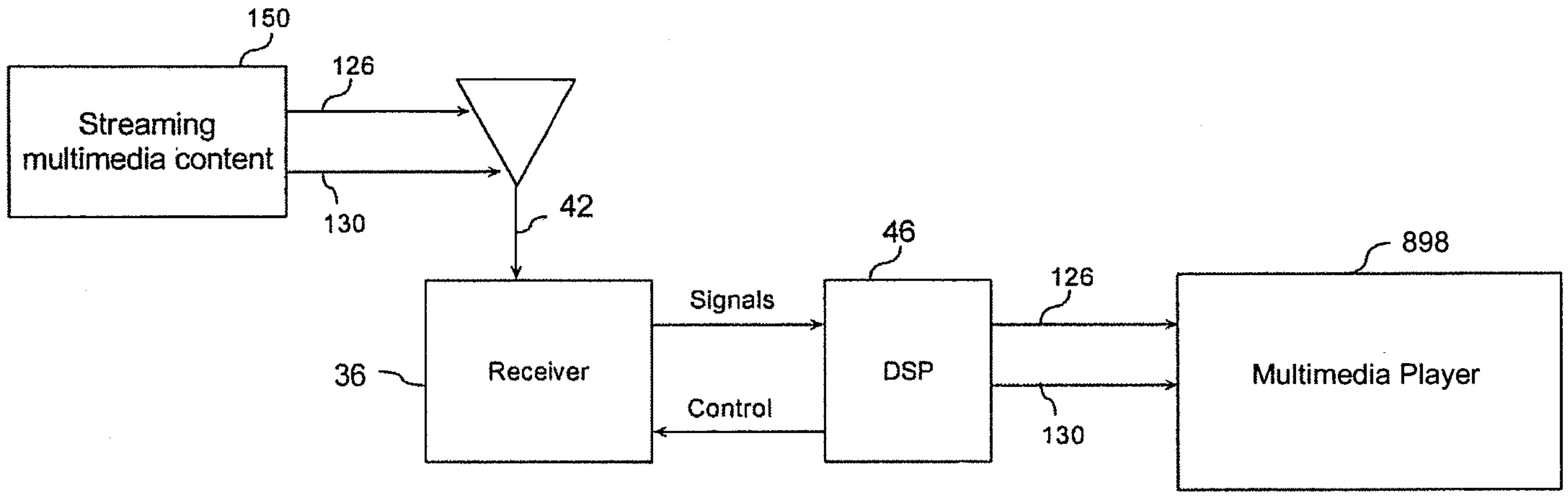

150
Streaming multimedia content
126
130
42
36
Receiver
Signals
Control
46
DSP
126
130
898
Multimedia Player

Figure 10

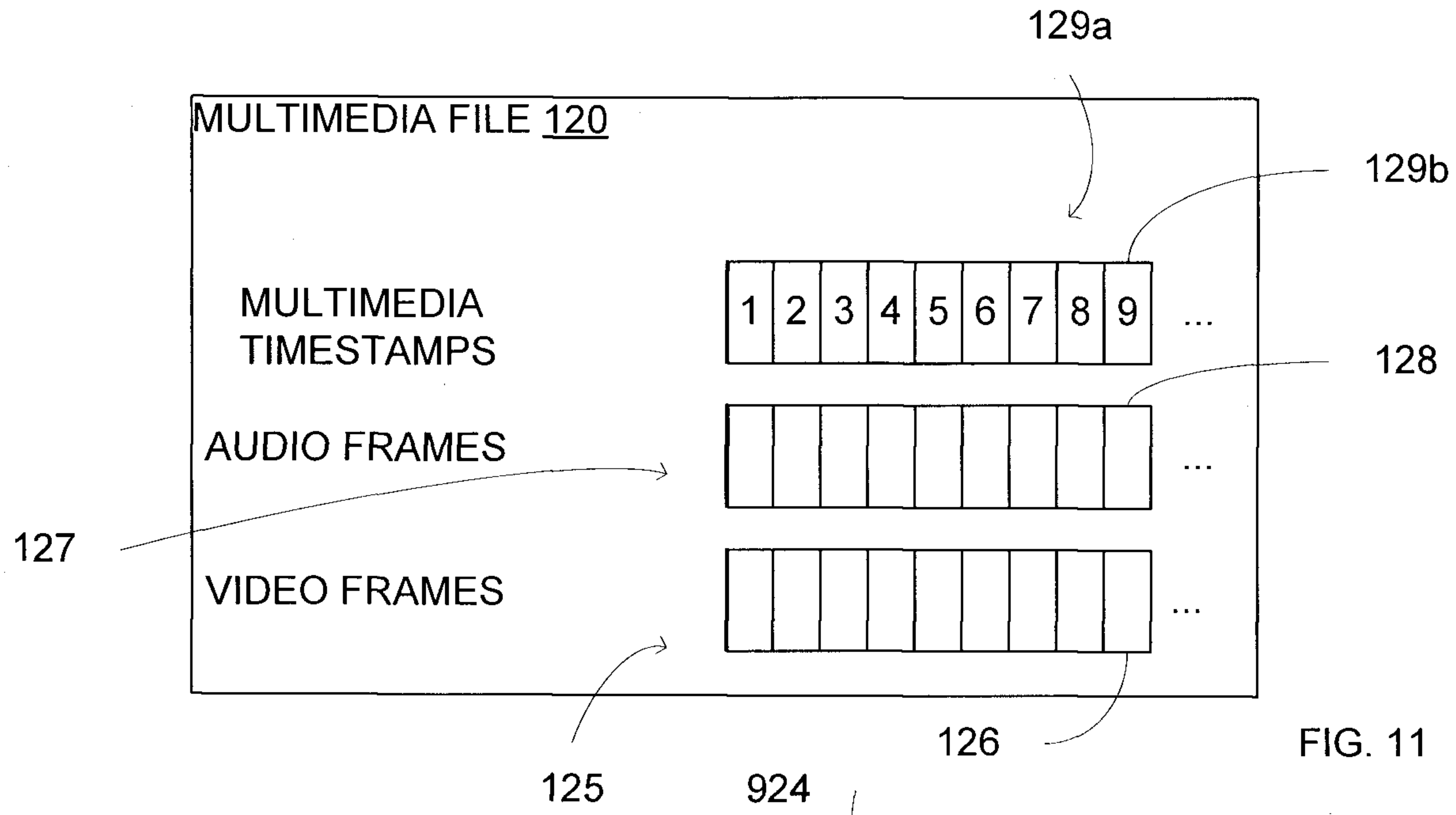


FIG. 11

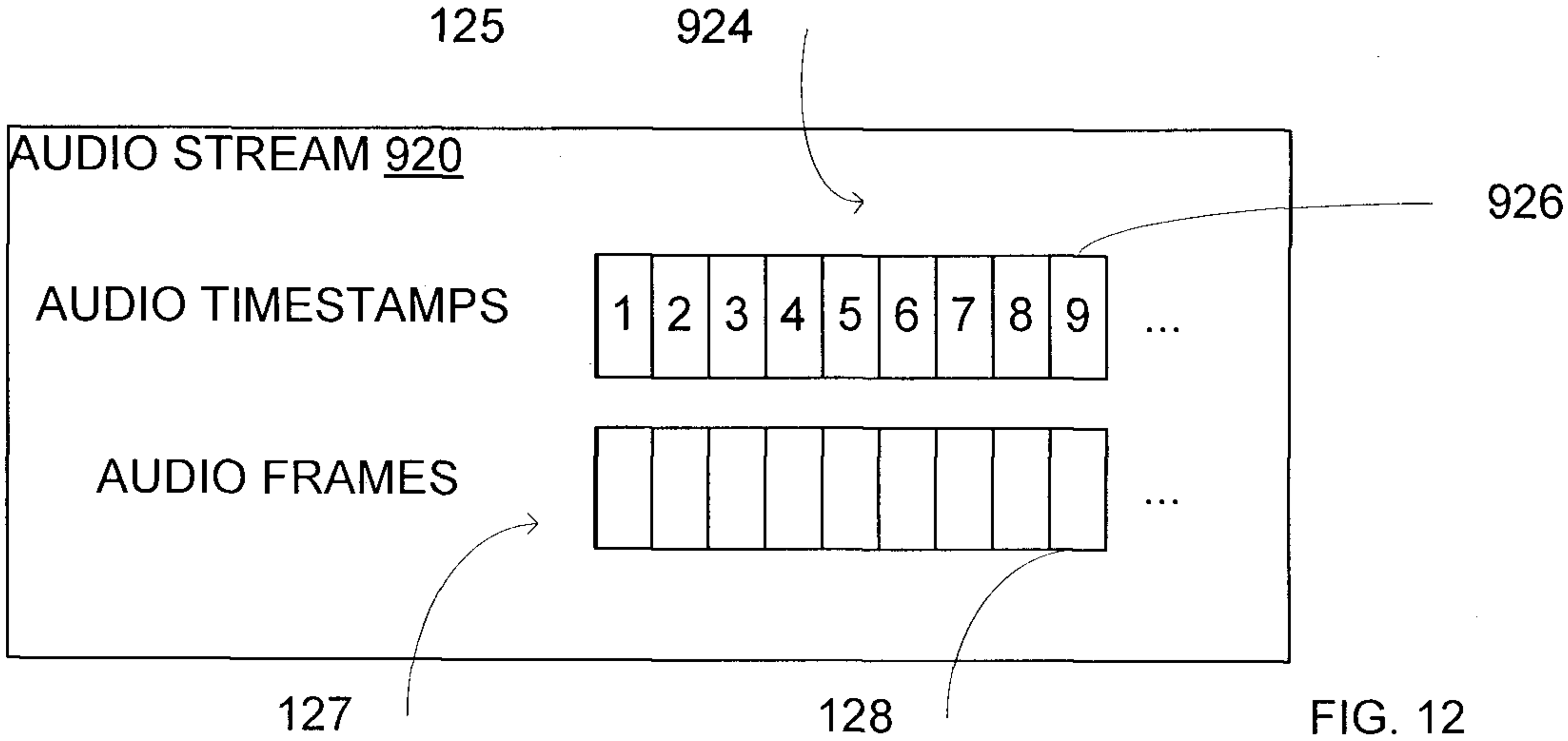


FIG. 12

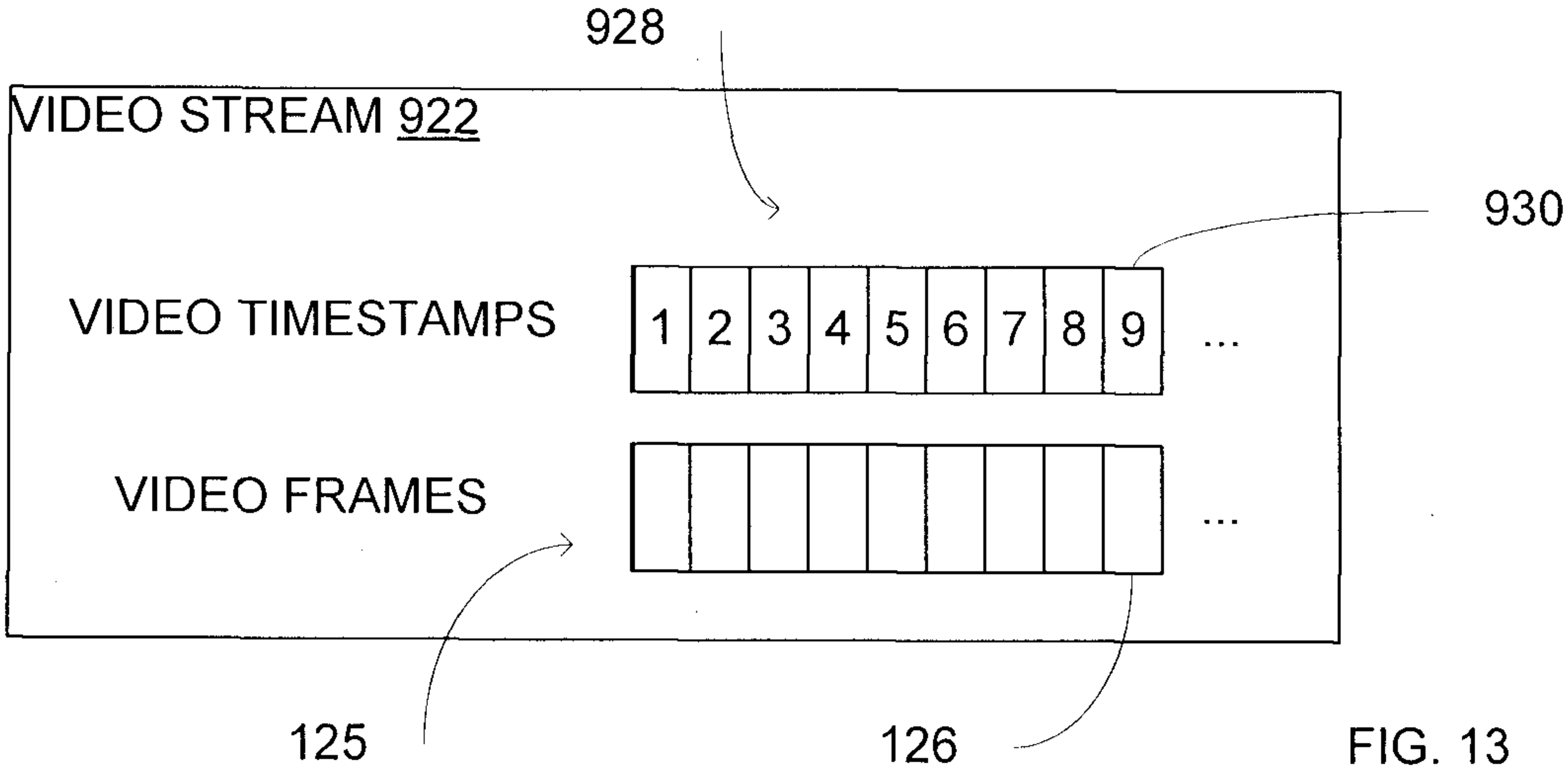


FIG. 13

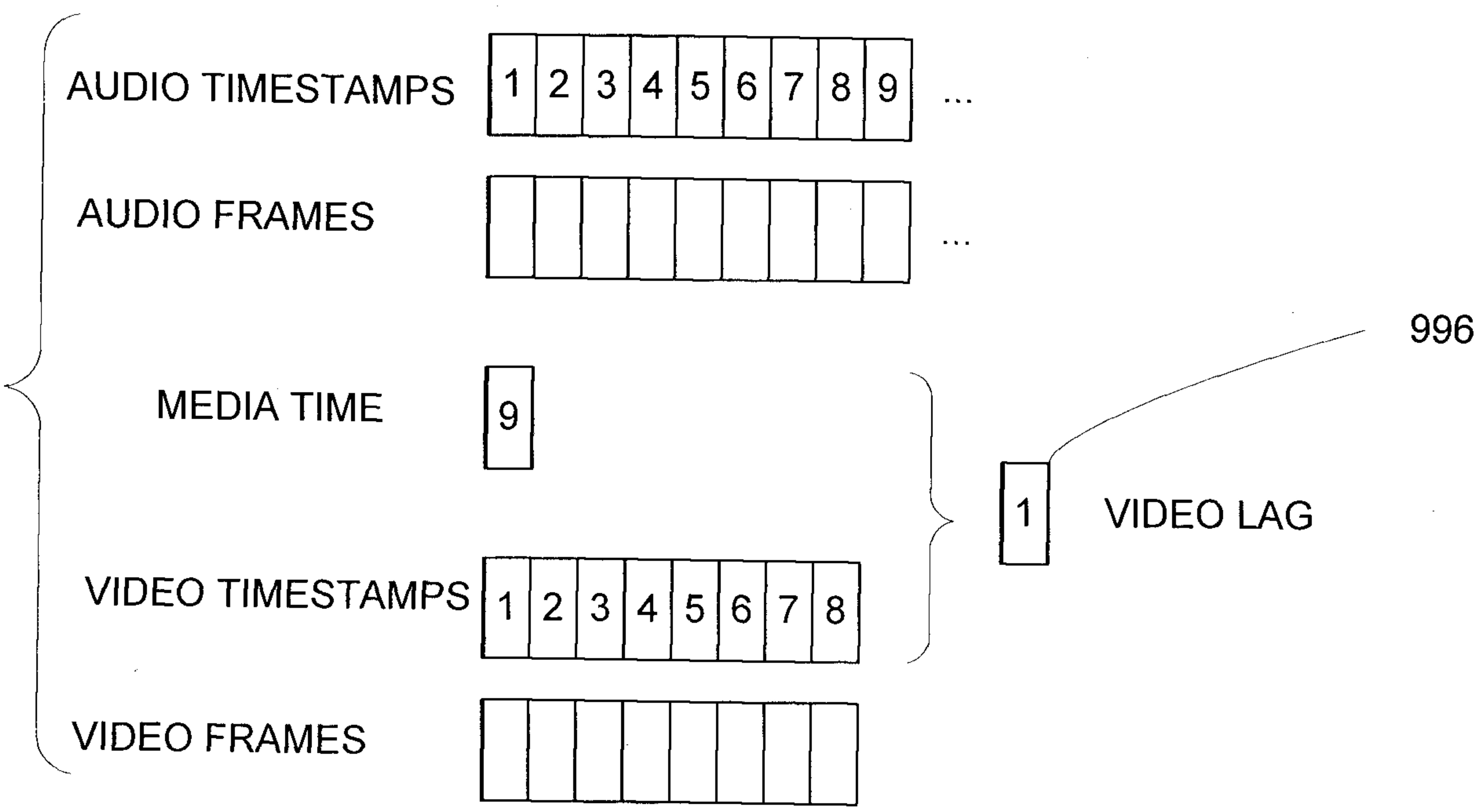

AUDIO TIMESTAMPS
1 2 3 4 5 6 7 8 9 ...
AUDIO FRAMES
...
996
MEDIA TIME
9
1
VIDEO LAG
VIDEO TIMESTAMPS
1 2 3 4 5 6 7 8
VIDEO FRAMES

FIG. 14

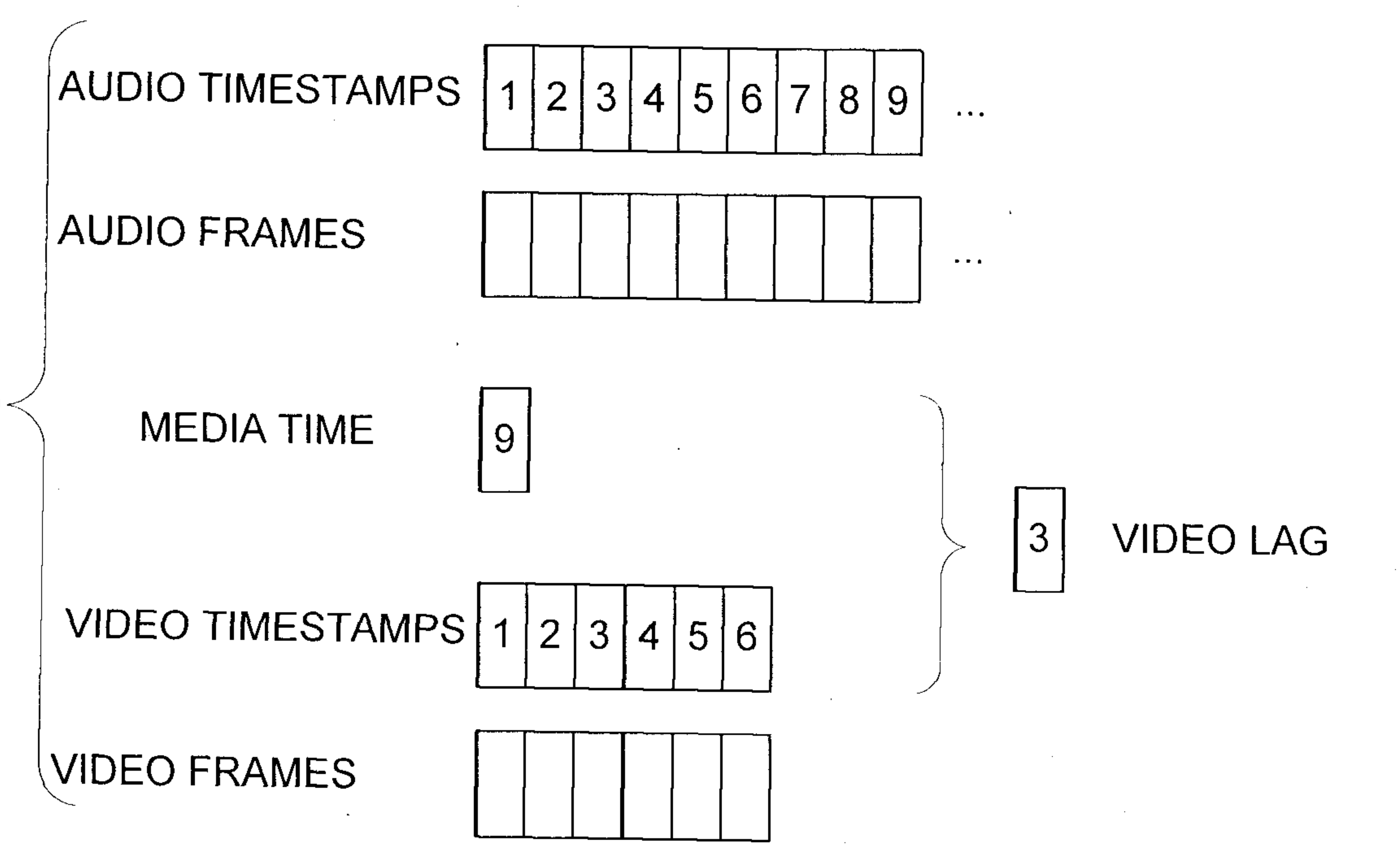

AUDIO TIMESTAMPS
1 2 3 4 5 6 7 8 9 ...
AUDIO FRAMES
...
MEDIA TIME
9
3
VIDEO LAG
VIDEO TIMESTAMPS
1 2 3 4 5 6
VIDEO FRAMES

FIG. 15

FIG. 16

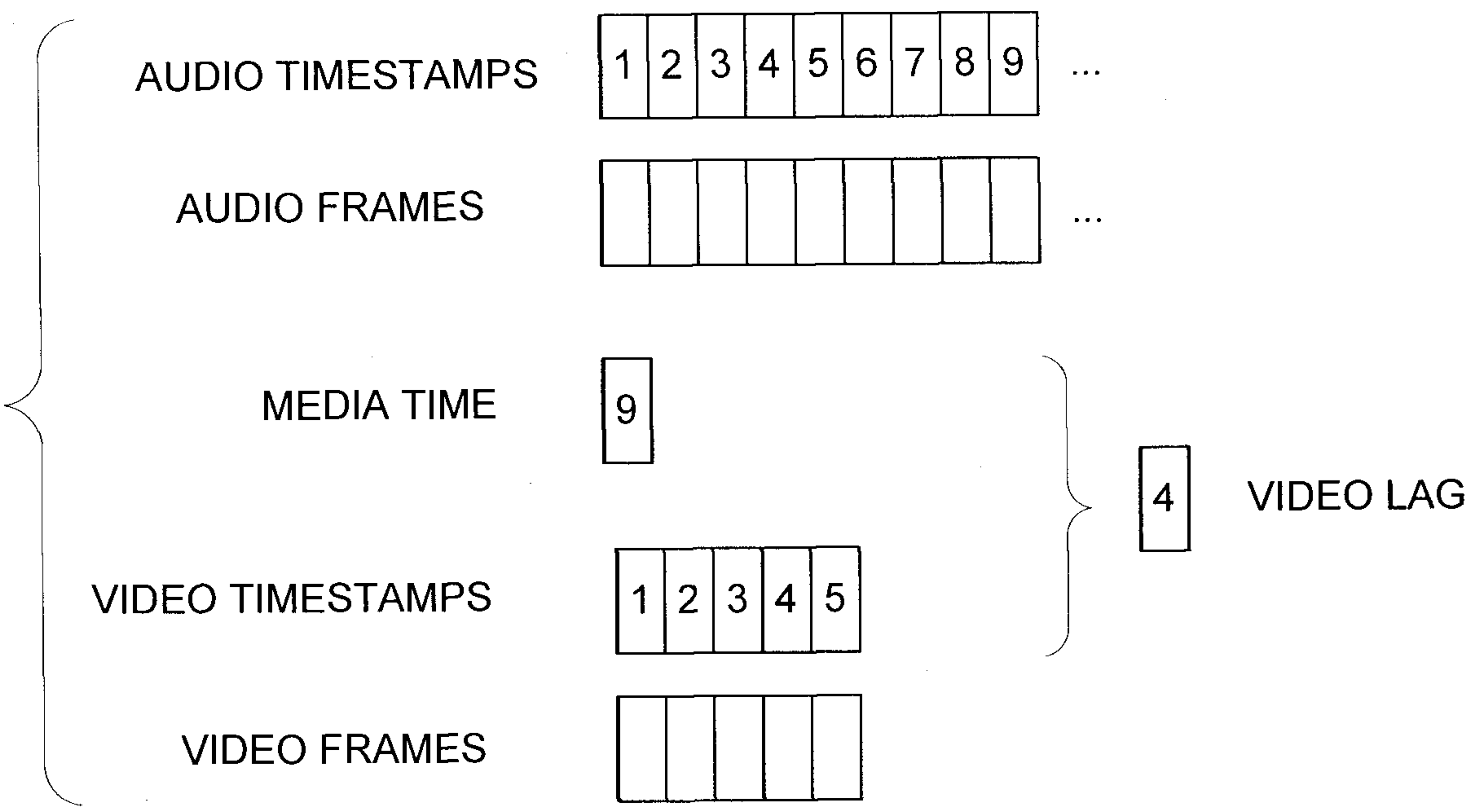

AUDIO TIMESTAMPS
1 2 3 4 5 6 7 8 9 ...
AUDIO FRAMES
...
MEDIA TIME
9
VIDEO TIMESTAMPS
1 2 3 4 5
VIDEO FRAMES
4
VIDEO LAG

FIG. 17

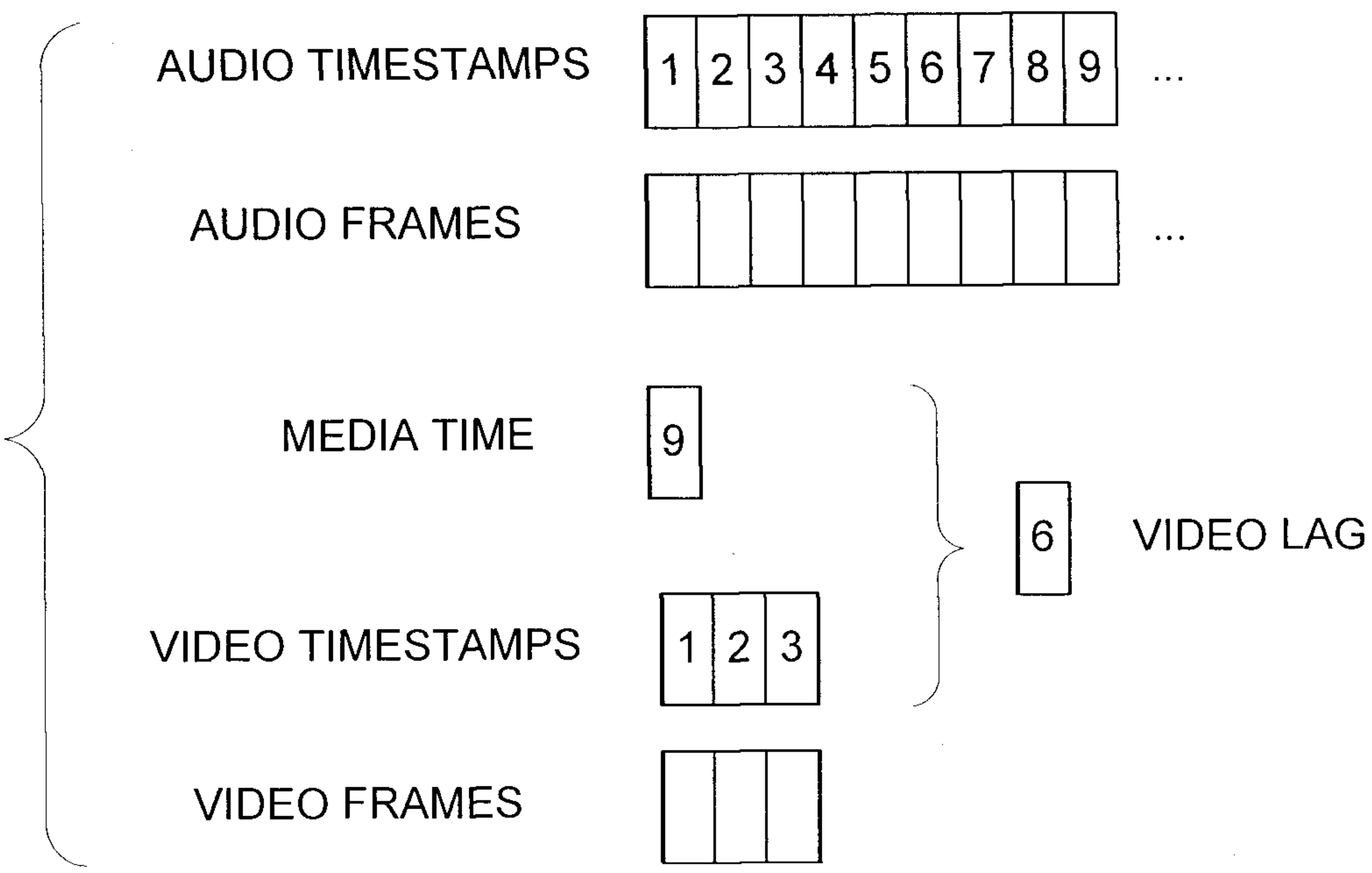
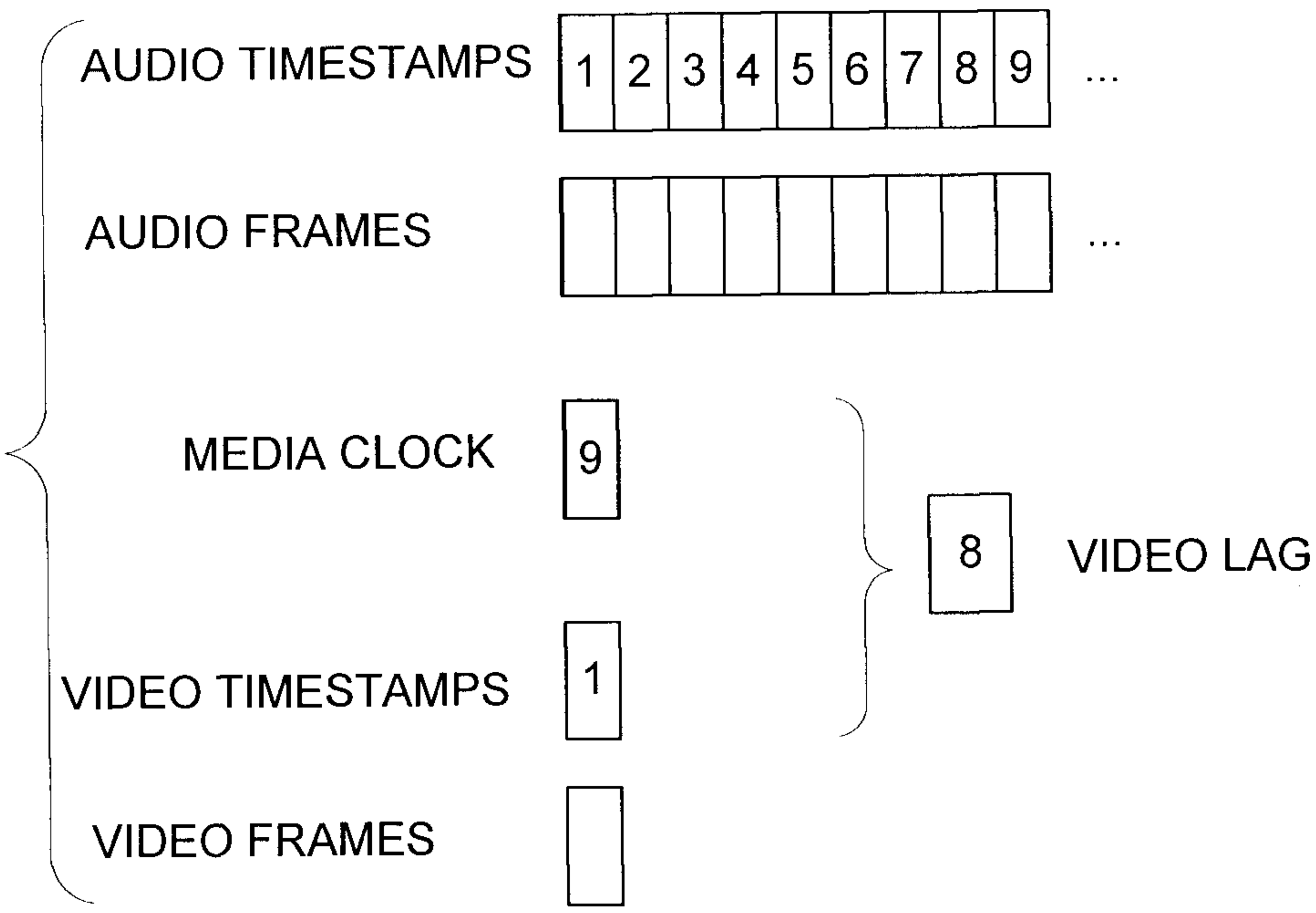

AUDIO TIMESTAMPS
1 2 3 4 5 6 7 8 9 ...
AUDIO FRAMES
...
MEDIA TIME
9
VIDEO TIMESTAMPS
1 2 3
VIDEO FRAMES
6
VIDEO LAG

FIG. 18


AUDIO TIMESTAMPS
1 2 3 4 5 6 7 8 9 ...
AUDIO FRAMES
...
MEDIA CLOCK
9
VIDEO TIMESTAMPS
1
VIDEO FRAMES
8
VIDEO LAG

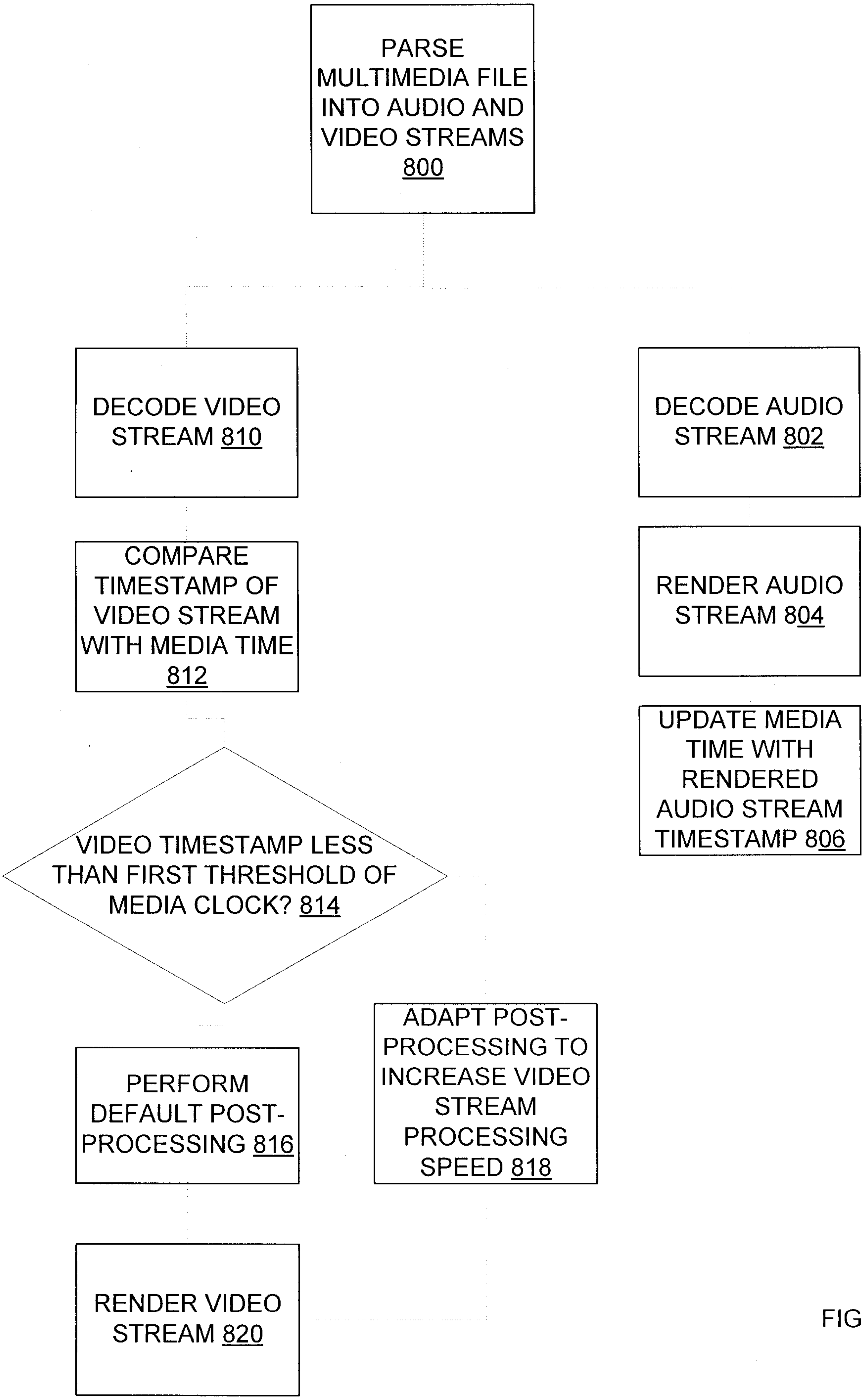

PARSE MULTIMEDIA FILE INTO AUDIO AND VIDEO STREAMS 800
DECODE VIDEO STREAM 810
COMPARE TIMESTAMP OF VIDEO STREAM WITH MEDIA TIME 812
VIDEO TIMESTAMP LESS THAN FIRST THRESHOLD OF MEDIA CLOCK? 814
PERFORM DEFAULT POST-PROCESSING 816
ADAPT POST-PROCESSING TO INCREASE VIDEO STREAM PROCESSING SPEED 818
RENDER VIDEO STREAM 820
DECODE AUDIO STREAM 802
RENDER AUDIO STREAM 804
UPDATE MEDIA TIME WITH RENDERED AUDIO STREAM TIMESTAMP 806

FIG. 19

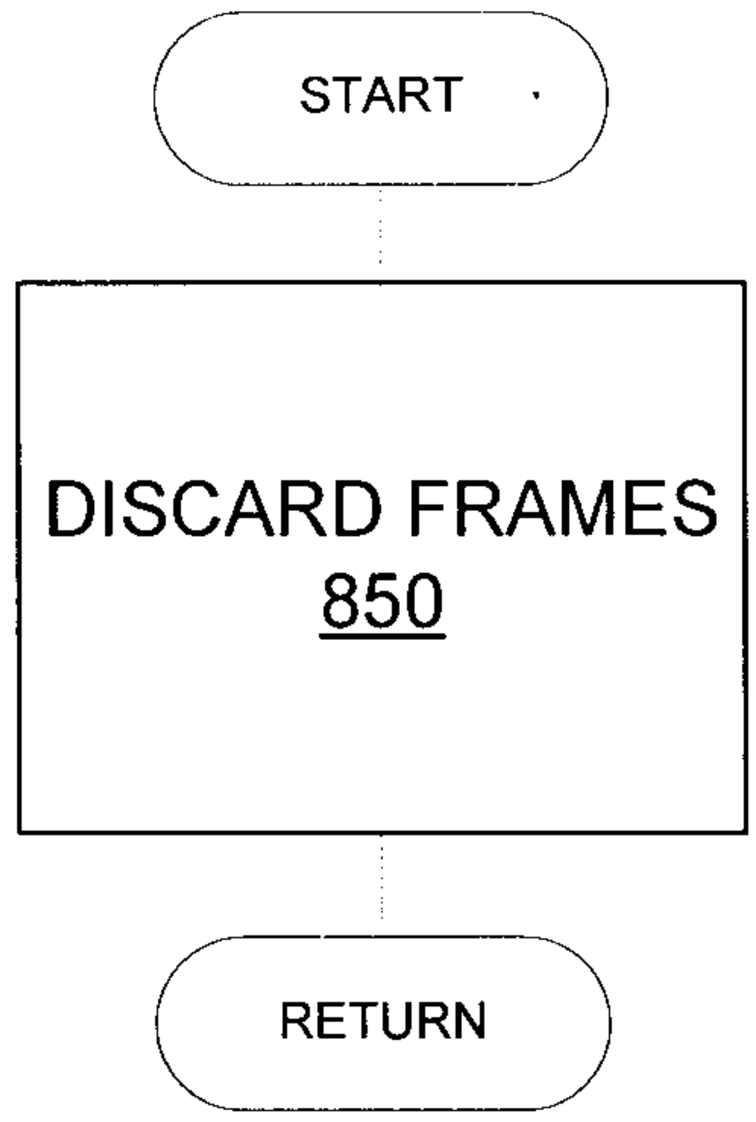


FIG. 20

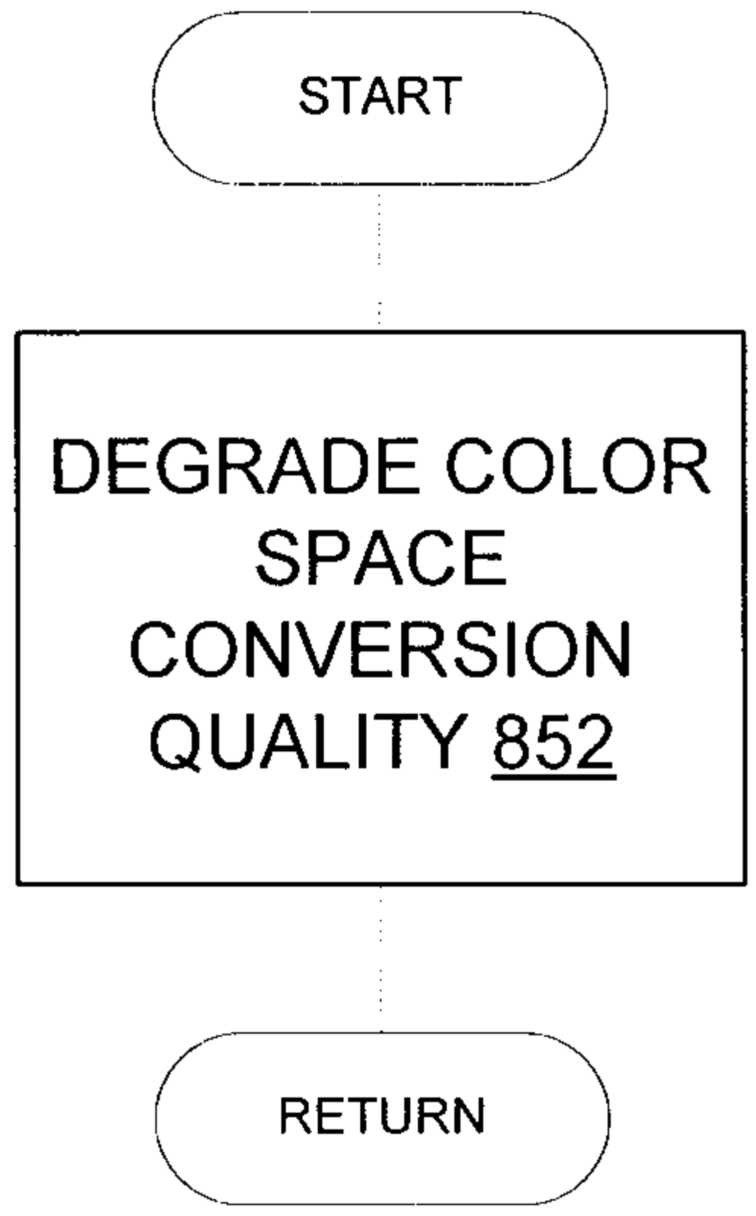


FIG. 21

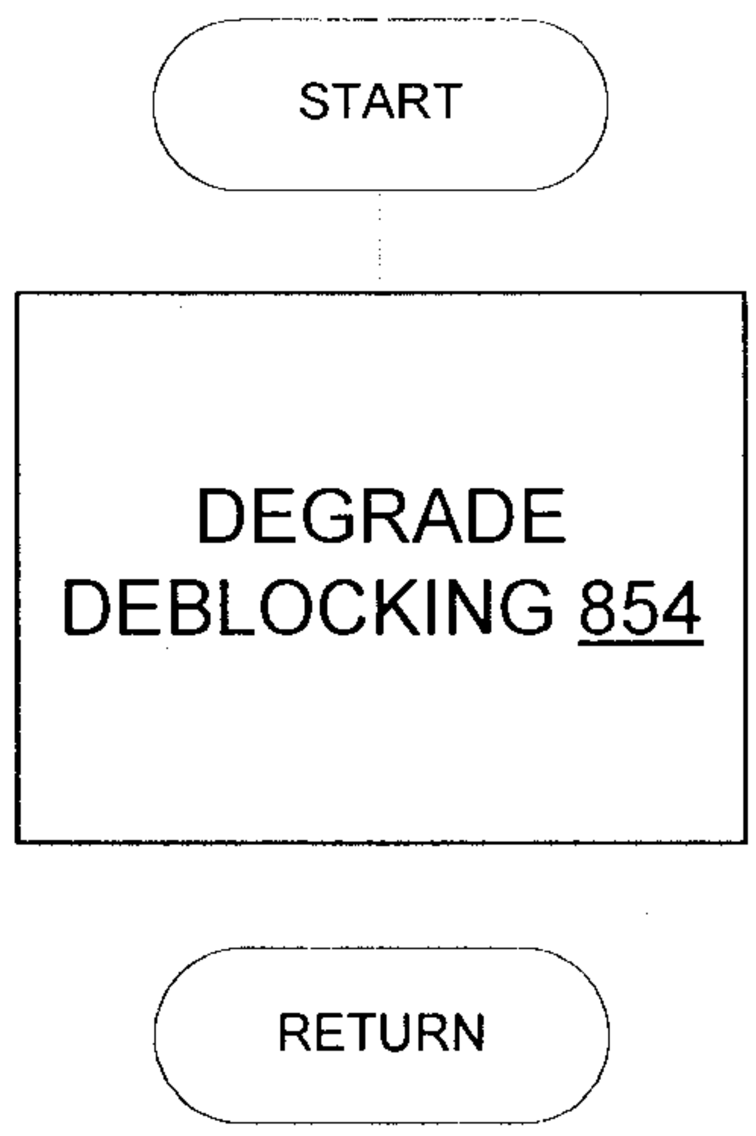


FIG. 22

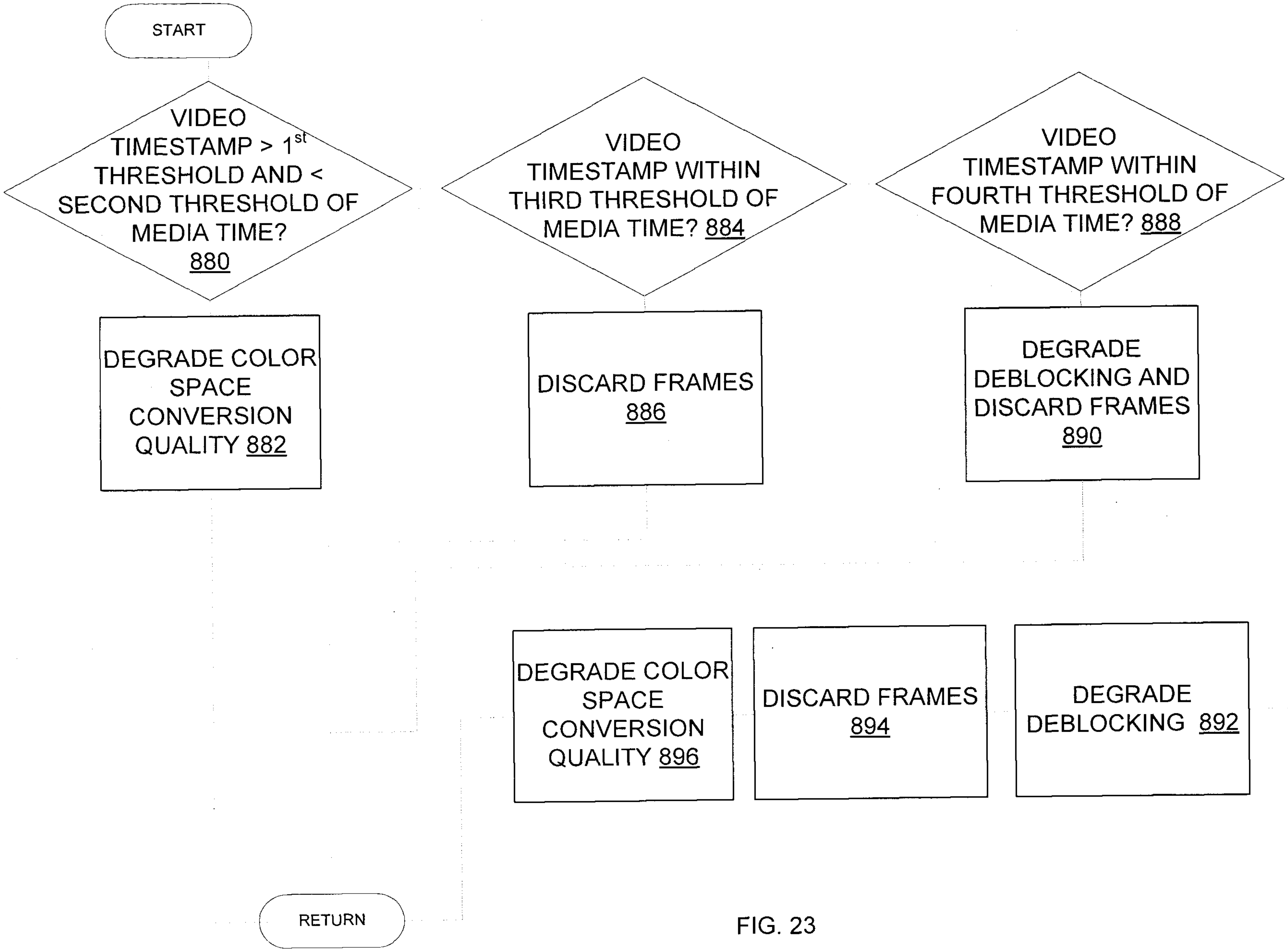

START
VIDEO TIMESTAMP > 1st THRESHOLD AND < SECOND THRESHOLD OF MEDIA TIME? 880
VIDEO TIMESTAMP WITHIN THIRD THRESHOLD OF MEDIA TIME? 884
VIDEO TIMESTAMP WITHIN FOURTH THRESHOLD OF MEDIA TIME? 888
DEGRADE COLOR SPACE CONVERSION QUALITY 882
DISCARD FRAMES 886
DEGRADE DEBLOCKING AND DISCARD FRAMES 890
DEGRADE COLOR SPACE CONVERSION QUALITY 896
DISCARD FRAMES 894
DEGRADE DEBLOCKING 892
RETURN

FIG. 23

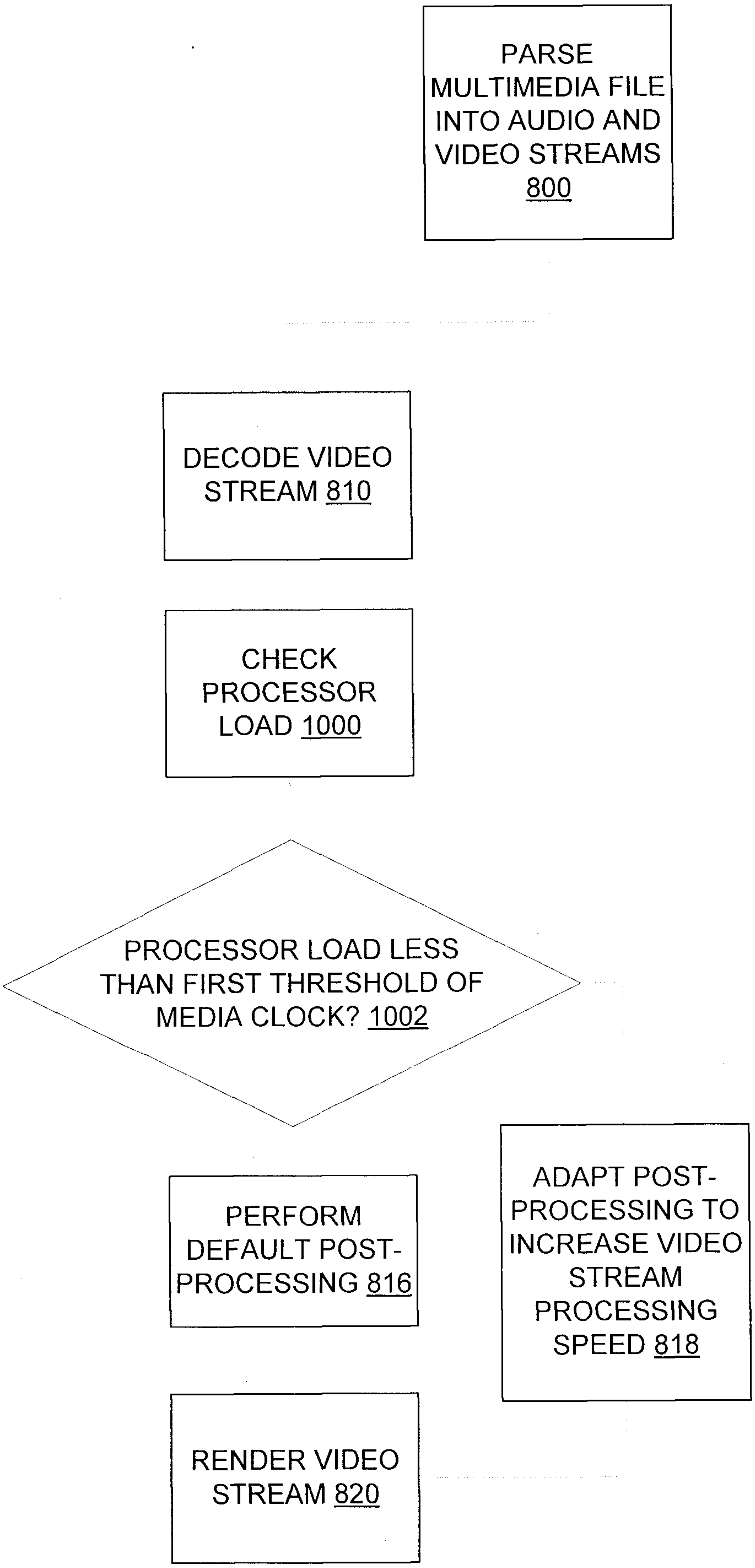
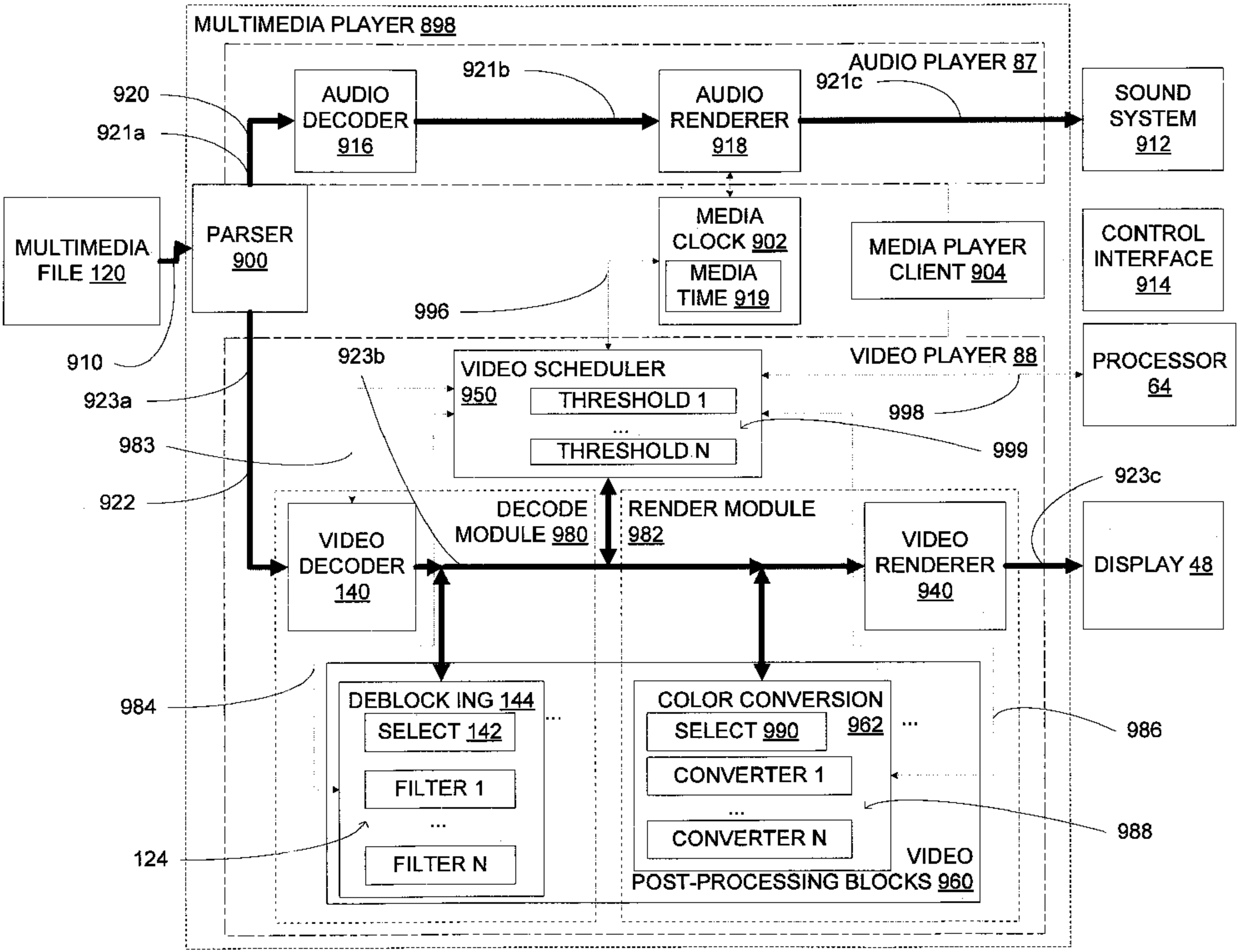

PARSE MULTIMEDIA FILE INTO AUDIO AND VIDEO STREAMS 800
DECODE VIDEO STREAM 810
CHECK PROCESSOR LOAD 1000
PROCESSOR LOAD LESS THAN FIRST THRESHOLD OF MEDIA CLOCK? 1002
PERFORM DEFAULT POST-PROCESSING 816
ADAPT POST-PROCESSING TO INCREASE VIDEO STREAM PROCESSING SPEED 818
RENDER VIDEO STREAM 820

FIG. 24


MULTIMEDIA PLAYER 898
AUDIO PLAYER 87
AUDIO DECODER 916
AUDIO RENDERER 918
SOUND SYSTEM 912
921b
921c
920
921a
MULTIMEDIA FILE 120
PARSER 900
MEDIA CLOCK 902
MEDIA TIME 919
MEDIA PLAYER CLIENT 904
CONTROL INTERFACE 914
996
910
923b
VIDEO PLAYER 88
PROCESSOR 64
VIDEO SCHEDULER 950
THRESHOLD 1
...
THRESHOLD N
923a
998
999
983
922
923c
DECODE MODULE 980
RENDER MODULE 982
VIDEO DECODER 140
VIDEO RENDERER 940
DISPLAY 48
984
DEBLOCK ING 144
SELECT 142
FILTER 1
...
FILTER N
COLOR CONVERSION 962
SELECT 990
CONVERTER 1
...
CONVERTER N
986
988
124
VIDEO POST-PROCESSING BLOCKS 960