US 20070110074A1

(19) **United States**

(12) **Patent Application Publication** (10) **Pub. No.: US 2007/0110074 A1**

Bradley et al. (43) **Pub. Date: May 17, 2007**

(54) **SYSTEM AND METHOD FOR SYNCHRONIZING MEDIA PRESENTATION AT MULTIPLE RECIPIENTS**

(76) Inventors: **Bob Bradley**, Foster City, CA (US); **Robert Dale Newberry JR.**, San Jose, CA (US)

Correspondence Address:
WONG, CABELLO, LUTSCH, RUTHERFORD & BRUCCULERI, L.L.P.
20333 SH 249
SUITE 600
HOUSTON, TX 77070 (US)

(21) Appl. No.: **11/306,557**

(22) Filed: **Jan. 2, 2006**

Related U.S. Application Data

(63) Continuation-in-part of application No. 10/862,115, filed on Jun. 4, 2004.

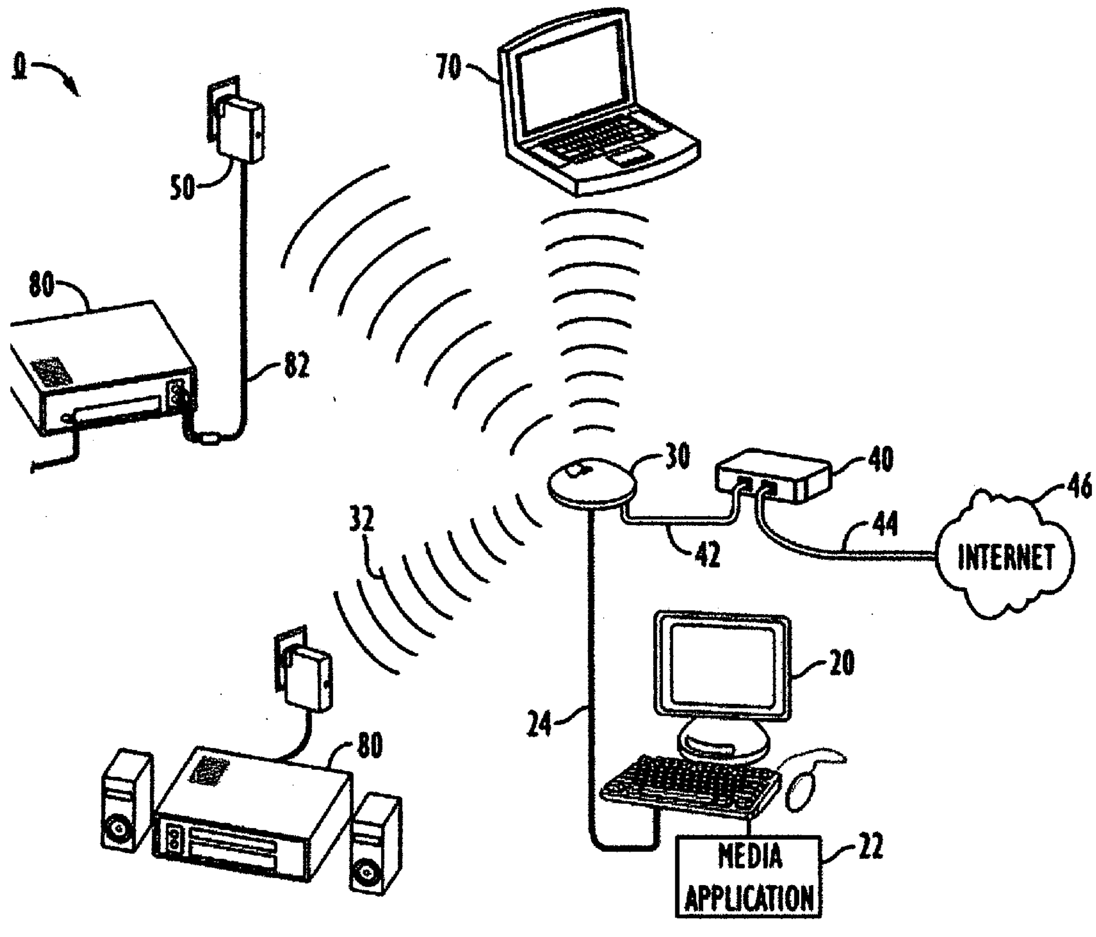
Publication Classification

(51) **Int. Cl.**
H04L 12/56 (2006.01)

(52) **U.S. Cl.** **370/395.51**

(57) **ABSTRACT**

A network media delivery system includes client devices and a host device. Each client device has a network interface, an engine for processing media data, and a media interface. The host device, which can be a computer, establishes network communication links with the client devices, which can be networked media stations, and sends media data to the client devices. The media data can be sent wirelessly as packets of media data transmitted at intervals to each client device. In one embodiment, the host device controls processing of media data such that processed media is delivered in a synchronized manner at each of the client devices. In another embodiment, the host device controls processing of media data such that processed media is delivered in a synchronized manner at the host device and at least one client device.



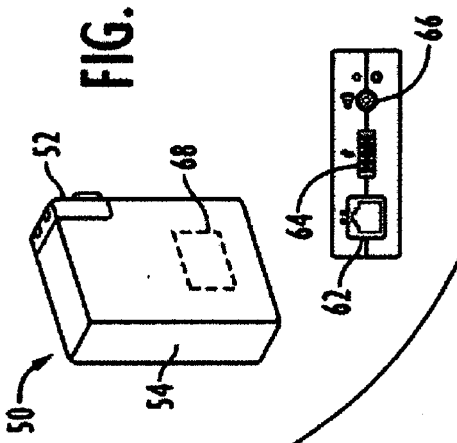

50
52
54
68
62
64
66

FIG. 2

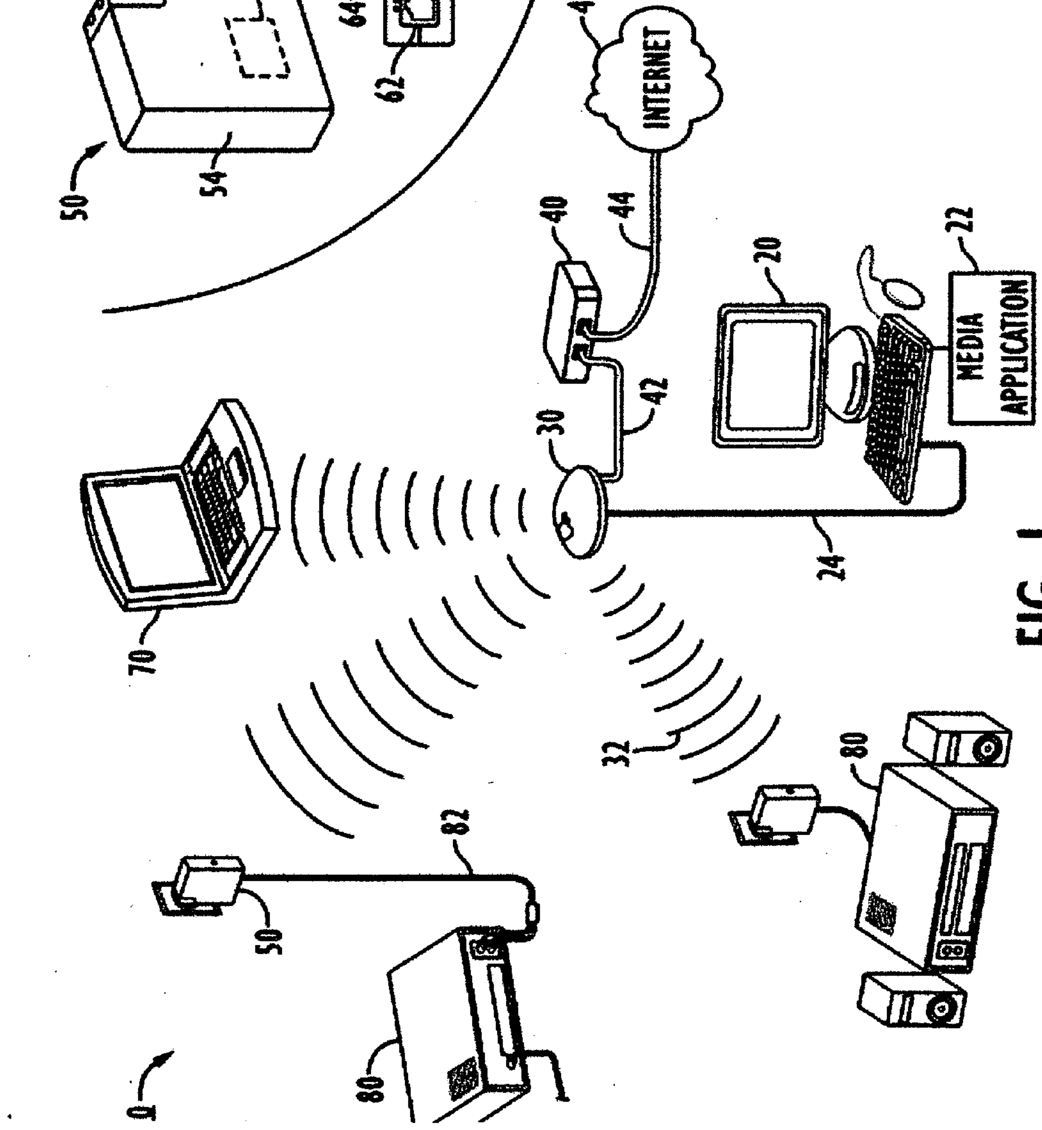

10
20
22
MEDIA APPLICATION
24
30
32
40
42
44
46
INTERNET
50
70
80
80
82

FIG. 1

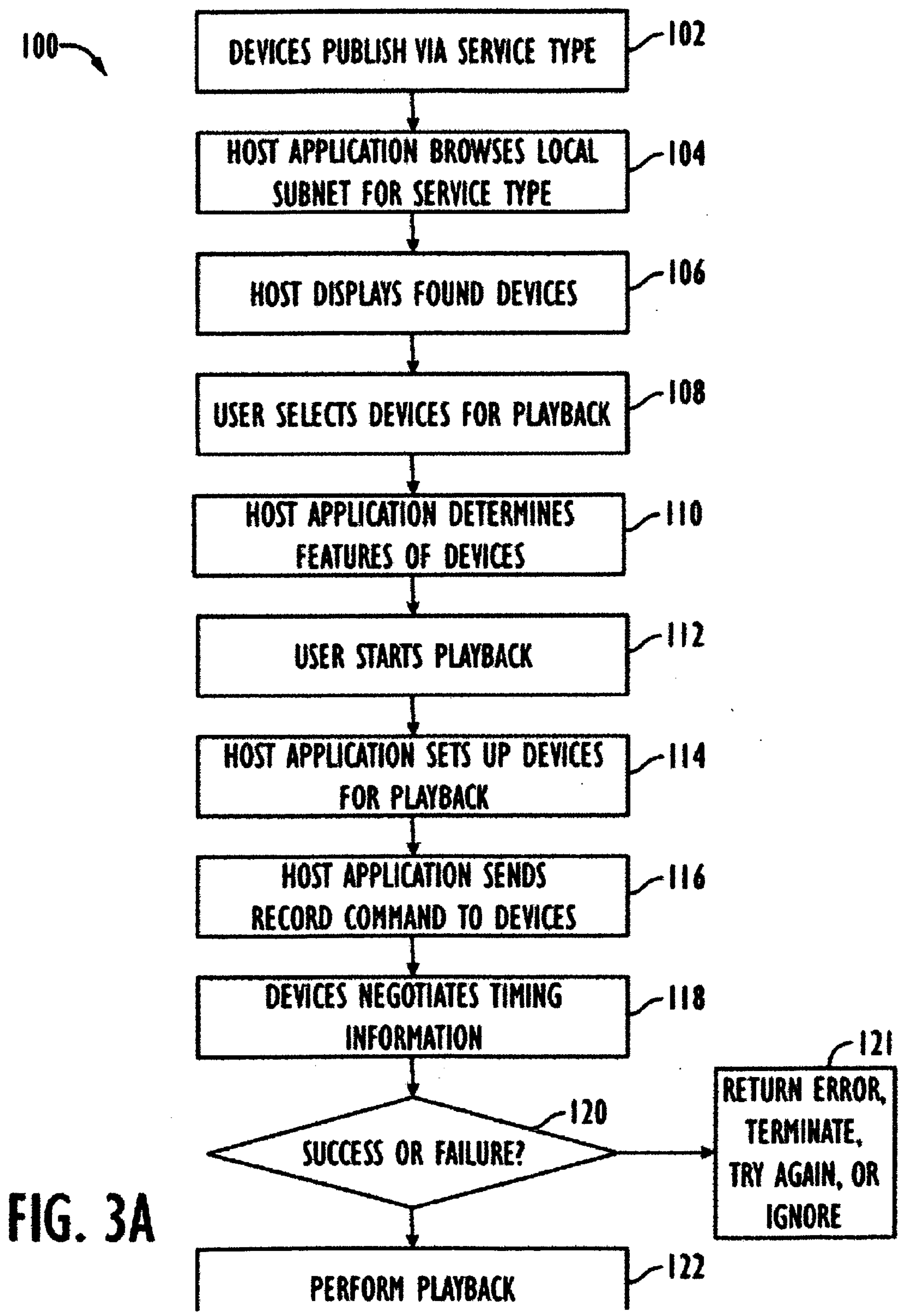

100
DEVICES PUBLISH VIA SERVICE TYPE
102
HOST APPLICATION BROWSES LOCAL SUBNET FOR SERVICE TYPE
104
HOST DISPLAYS FOUND DEVICES
106
USER SELECTS DEVICES FOR PLAYBACK
108
HOST APPLICATION DETERMINES FEATURES OF DEVICES
110
USER STARTS PLAYBACK
112
HOST APPLICATION SETS UP DEVICES FOR PLAYBACK
114
HOST APPLICATION SENDS RECORD COMMAND TO DEVICES
116
DEVICES NEGOTIATES TIMING INFORMATION
118
SUCCESS OR FAILURE?
120
RETURN ERROR, TERMINATE, TRY AGAIN, OR IGNORE
121
PERFORM PLAYBACK
122

FIG. 3A

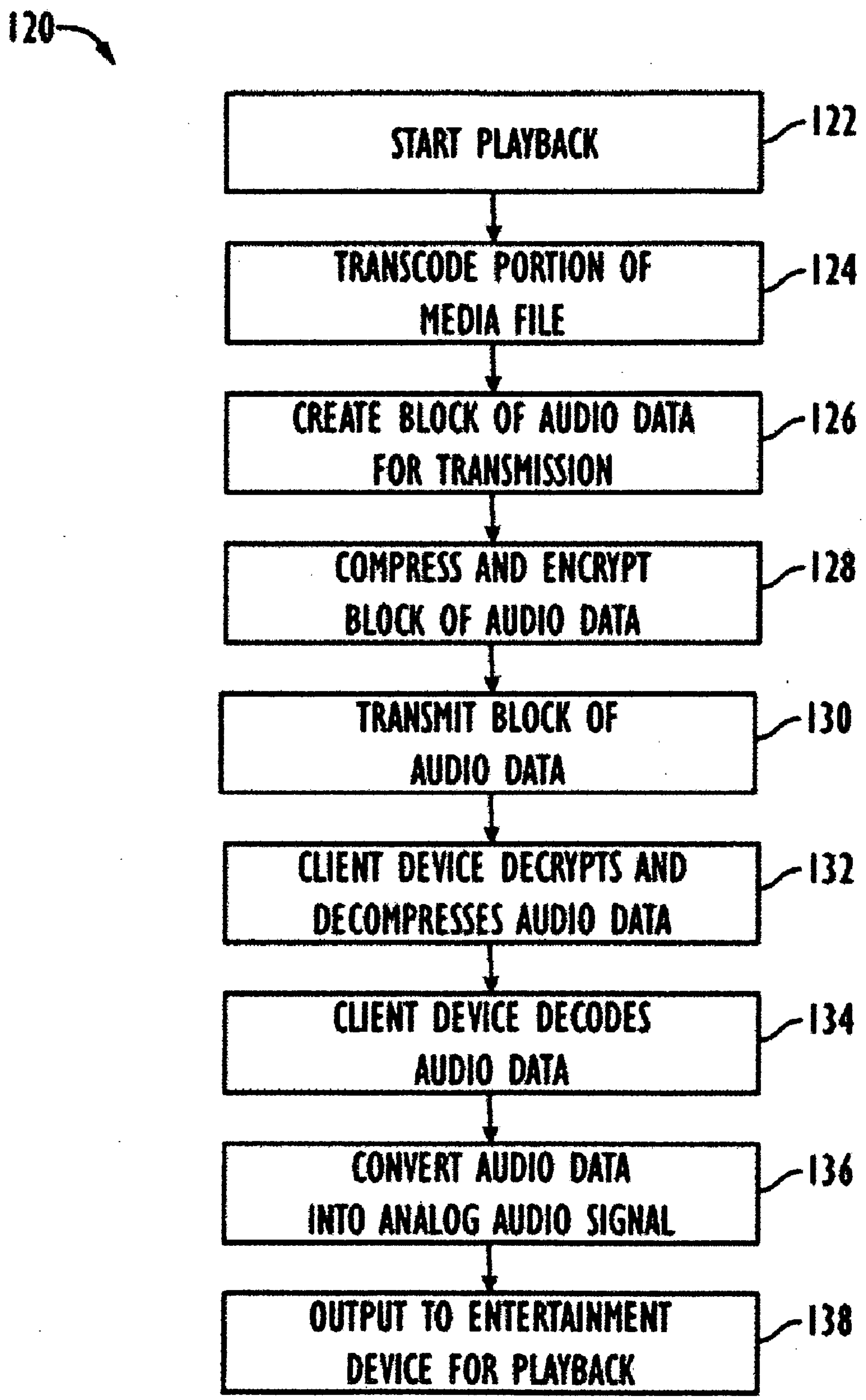
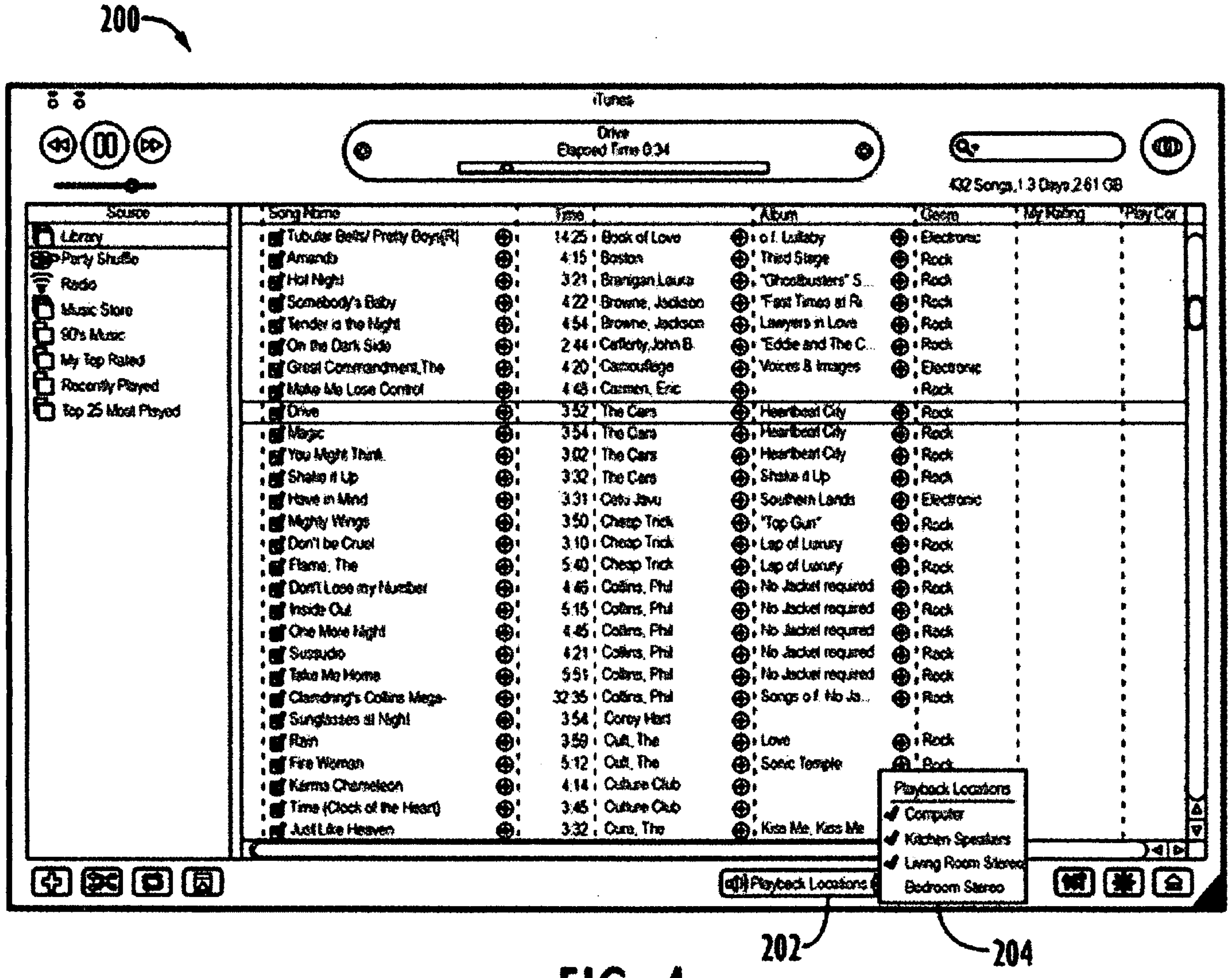

120
START PLAYBACK
122
TRANSCODE PORTION OF MEDIA FILE
124
CREATE BLOCK OF AUDIO DATA FOR TRANSMISSION
126
COMPRESS AND ENCRYPT BLOCK OF AUDIO DATA
128
TRANSMIT BLOCK OF AUDIO DATA
130
CLIENT DEVICE DECRYPTS AND DECOMPRESSES AUDIO DATA
132
CLIENT DEVICE DECODES AUDIO DATA
134
CONVERT AUDIO DATA INTO ANALOG AUDIO SIGNAL
136
OUTPUT TO ENTERTAINMENT DEVICE FOR PLAYBACK
138

FIG. 3B



Song Name	Time		Album	Genre	My Rating	Play Cou
Tubular Bells/ Pretty Boys[R]	14:25	Book of Love	o.f. Lullaby	Electronic		
Amanda	4:15	Boston	Third Stage	Rock		
Hot Night	3:21	Branigan, Laura	"Ghostbusters" S...	Rock		
Somebody's Baby	4:22	Browne, Jackson	"Fast Times at R	Rock		
Tender is the Night	4:54	Browne, Jackson	Lawyers in Love	Rock		
On the Dark Side	2:44	Cafferty, John B.	"Eddie and The C...	Rock		
Great Commandment, The	4:20	Camouflage	Voices & Images	Electronic		
Make Me Lose Control	4:48	Carmen, Eric		Rock		
Drive	3:52	The Cars	Heartbeat City	Rock		
Magic	3:54	The Cars	Heartbeat City	Rock		
You Might Think	3:02	The Cars	Heartbeat City	Rock		
Shake it Up	3:32	The Cars	Shake it Up	Rock		
Have in Mind	3:31	Ceto Javu	Southern Lands	Electronic		
Mighty Wings	3:50	Cheap Trick	"Top Gun"	Rock		
Don't be Cruel	3:10	Cheap Trick	Lap of Luxury	Rock		
Flame, The	5:40	Cheap Trick	Lap of Luxury	Rock		
Don't Lose my Number	4:46	Collins, Phil	No Jacket required	Rock		
Inside Out	5:15	Collins, Phil	No Jacket required	Rock		
One More Night	4:46	Collins, Phil	No Jacket required	Rock		
Sussudio	4:21	Collins, Phil	No Jacket required	Rock		
Take Me Home	5:51	Collins, Phil	No Jacket required	Rock		
Clamdrang's Collins Mega-	32:35	Collins, Phil	Songs o f No Ja...	Rock		
Sunglasses at Night	3:54	Corey Hart				
Rain	3:59	Cult, The	Love	Rock		
Fire Woman	5:12	Cult, The	Sonic Temple	Rock		
Karma Chameleon	4:14	Culture Club				
Time (Clock of the Heart)	3:45	Culture Club				
Just Like Heaven	3:32	Cure, The	Kiss Me, Kiss Me			

FIG. 4

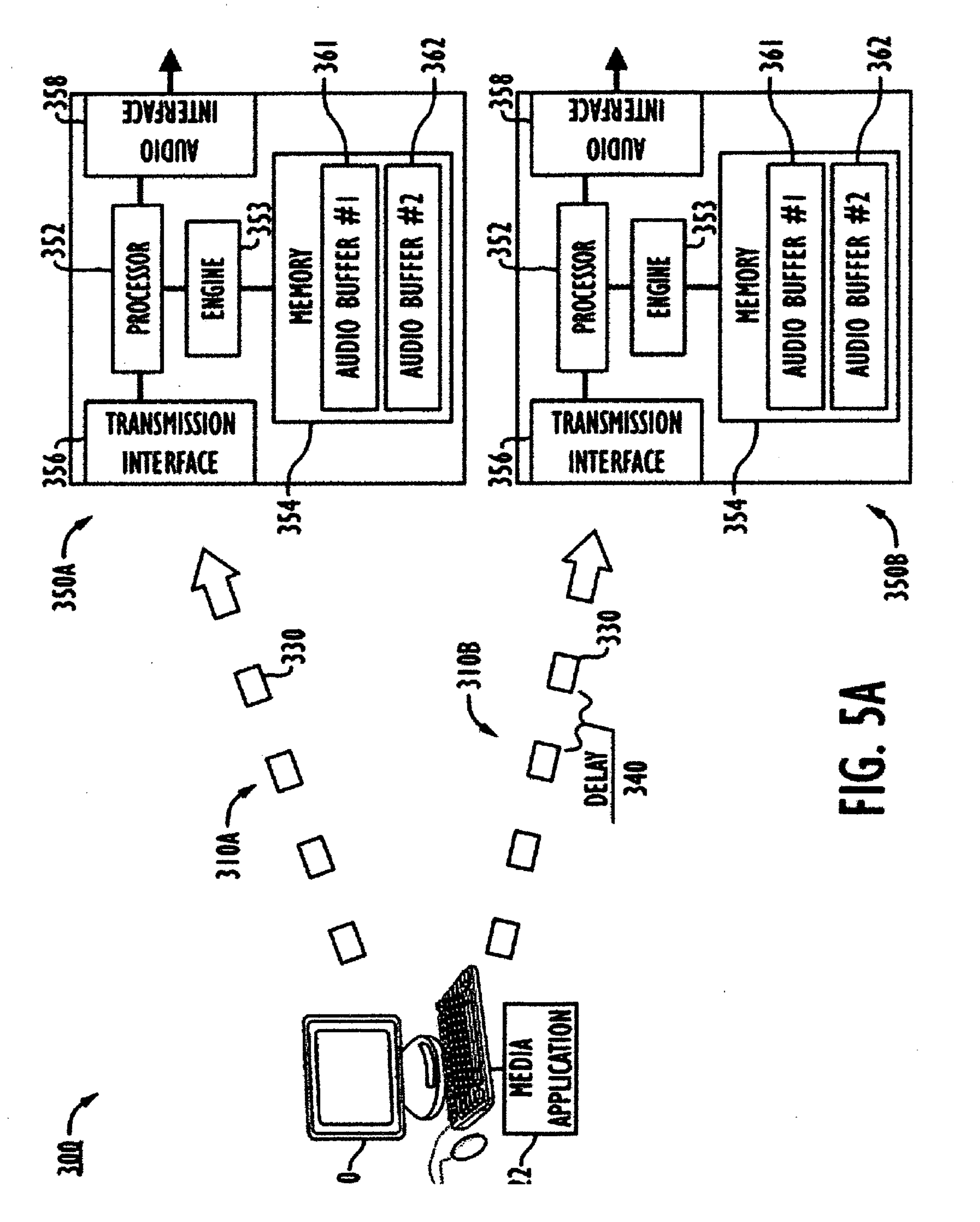

300
0
12
MEDIA APPLICATION
310A
310B
330
330
DELAY
340
350A
350B
TRANSMISSION INTERFACE
PROCESSOR
ENGINE
MEMORY
AUDIO BUFFER #1
AUDIO BUFFER #2
AUDIO INTERFACE
352
353
354
356
358
361
362

FIG. 5A

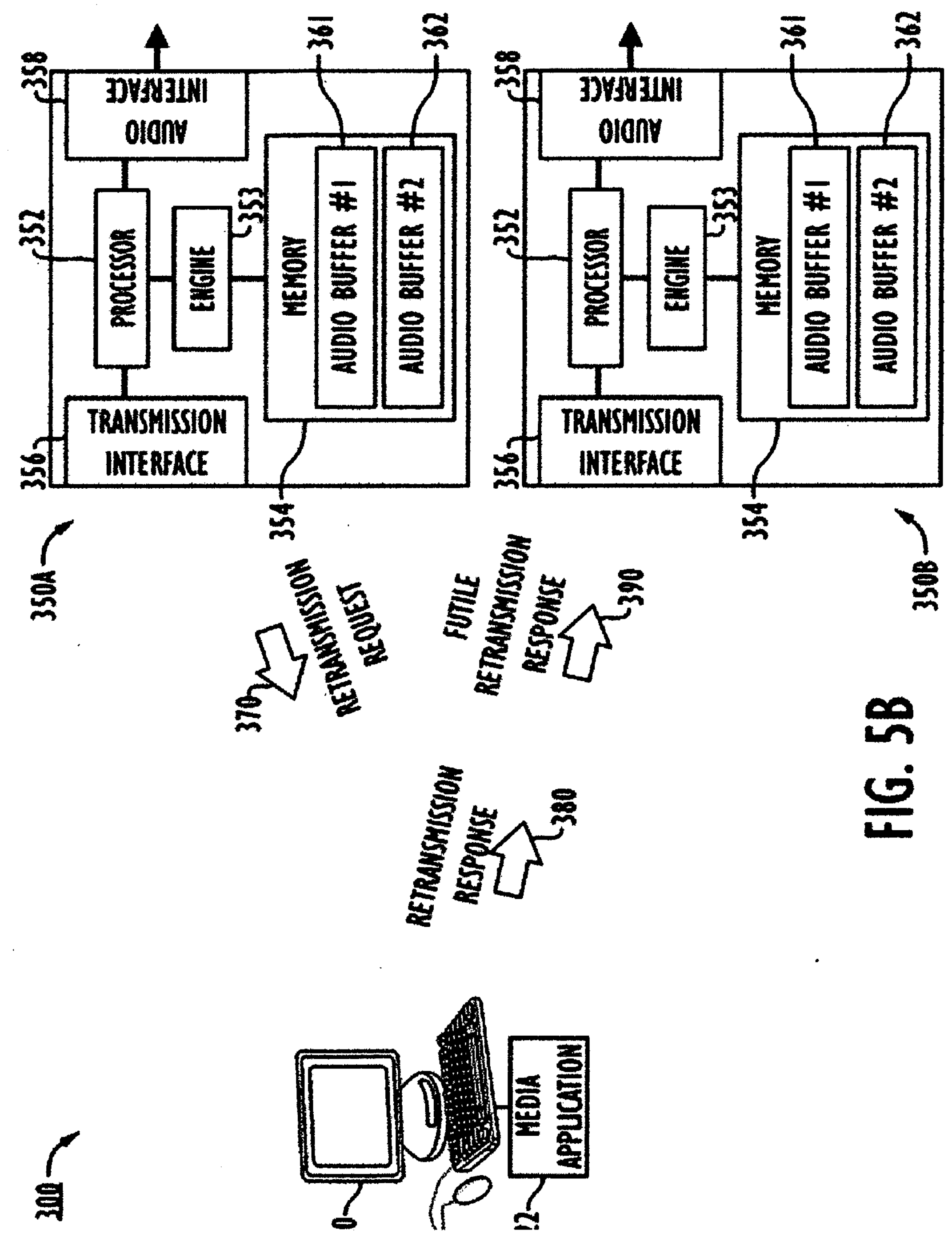

300
0
MEDIA APPLICATION
12
RETRANSMISSION RESPONSE
380
RETRANSMISSION REQUEST
370
FUTILE RETRANSMISSION RESPONSE
390
350A
350B
TRANSMISSION INTERFACE
356
PROCESSOR
352
ENGINE
353
MEMORY
354
AUDIO BUFFER #1
361
AUDIO BUFFER #2
362
AUDIO INTERFACE
358
TRANSMISSION INTERFACE
356
PROCESSOR
352
ENGINE
353
MEMORY
354
AUDIO BUFFER #1
361
AUDIO BUFFER #2
362
AUDIO INTERFACE
358

FIG. 5B

FIG. 6A

370

```
                       RTCP RETRANSMIT REQUEST PACKET
                         1                   2                   3
               0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
              +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +0/0X00
HEADER        |V=2|P|    0    |    PT=213     |             LENGTH            |
ETRANSMIT     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +4/0X04
IFORMATION    |     SEQUENCE NUMBER BASE      |     SEQUENCE NUMBER COUNT     |
              +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +8/0X08
```

FIG. 6B

380

```
                       RTCP RETRANSMIT RESPONSE PACKET
                         1                   2                   3
               0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
              +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +0/0X00
HEADER        |V=2|P|    0    |    PT=214     |             LENGTH            |
ETRANSMIT     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +4/0X0C
FORMATION     |                                                               |
              |                            PAYLOAD                            |
              |                                                               |
              +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +1444/0X5A4
```

FIG. 6C

390

```
                    RTCP FUTILE RETRANSMIT RESPONSE PACKET
                         1                   2                   3
               0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
              +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +0/0X00
HEADER        |V=2|P|    0    |    PT=214     |             LENGTH            |
ETRANSMIT     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +4/0X0C
IFORMATION    |     SEQUENCE NUMBER BASE      |              PAD              |
              +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +1444/0X5A4
```

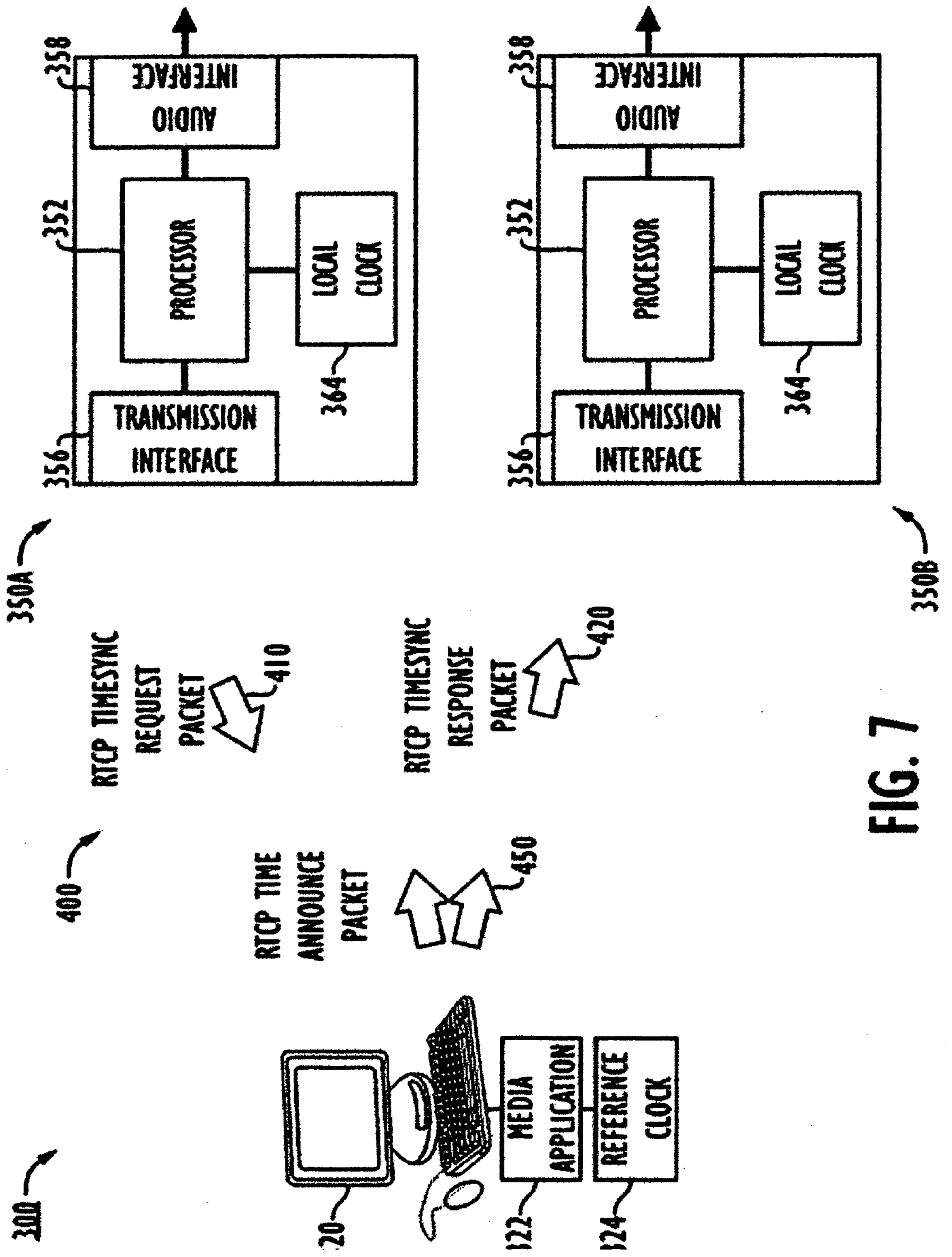

300
20
122
MEDIA APPLICATION
124
REFERENCE CLOCK
400
RTCP TIME ANNOUNCE PACKET
450
RTCP TIMESYNC REQUEST PACKET
410
RTCP TIMESYNC RESPONSE PACKET
420
350A
350B
356
TRANSMISSION INTERFACE
352
PROCESSOR
364
LOCAL CLOCK
358
AUDIO INTERFACE

FIG. 7

430

```
                             RTCP TIMESYNC PACKET
                      1                   2                   3
        0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +0/0X00
HEADER |V=2|P|    0    |     PT=214    |             LENGTH            |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +4/0X04
TIMING |          RTP TIMESTAMP AT NTP TRANSMIT (T3) TIME              |
FORMATION
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +8/0X08
       |          NTP ORIGINATE (T1) TIMESTAMP, MOST SIGNIFICANT WORD  |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +12/0X0C
       |          NTP ORIGINATE (T1) TIMESTAMP, LEAST SIGNIFICANT WORD |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +16/0X10
       |          NTP RECEIVE (T2) TIMESTAMP, MOST SIGNIFICANT WORD    |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +20/0X14
       |          NTP RECEIVE (T2) TIMESTAMP, LEAST SIGNIFICANT WORD   |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +24/0X18
       |          NTP TRANSMIT (T3) TIMESTAMP, MOST SIGNIFICANT WORD   |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +28/0X1C
       |          NTP TRANSMIT (T3) TIMESTAMP, LEAST SIGNIFICANT WORD  |
       +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +32/0X20
```

FIG. 8A

```
 450
                        RTCP TIME ANNOUNCE PACKET
                     1                   2                   3
                 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +0/0X00
HEADER          |V=2|P|    0    |    PT=212     |            LENGTH             |
                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +4/0X04
TIMING          |      RTP TIMESTAMP                                            |
INFORMATION     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +8/0X08
                |      NTP TIMESTAMP, HIGH 32 BITS                              |
                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +12/0X0C
                |      NTP TIMESTAMP, LOW 32 BITS                               |
                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +16/0X10
                |      RTP TIMESTAMP WHEN NEW TIMELINE SHOULD BE APPLIED        |
                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ +20/0X14
```

FIG. 8B

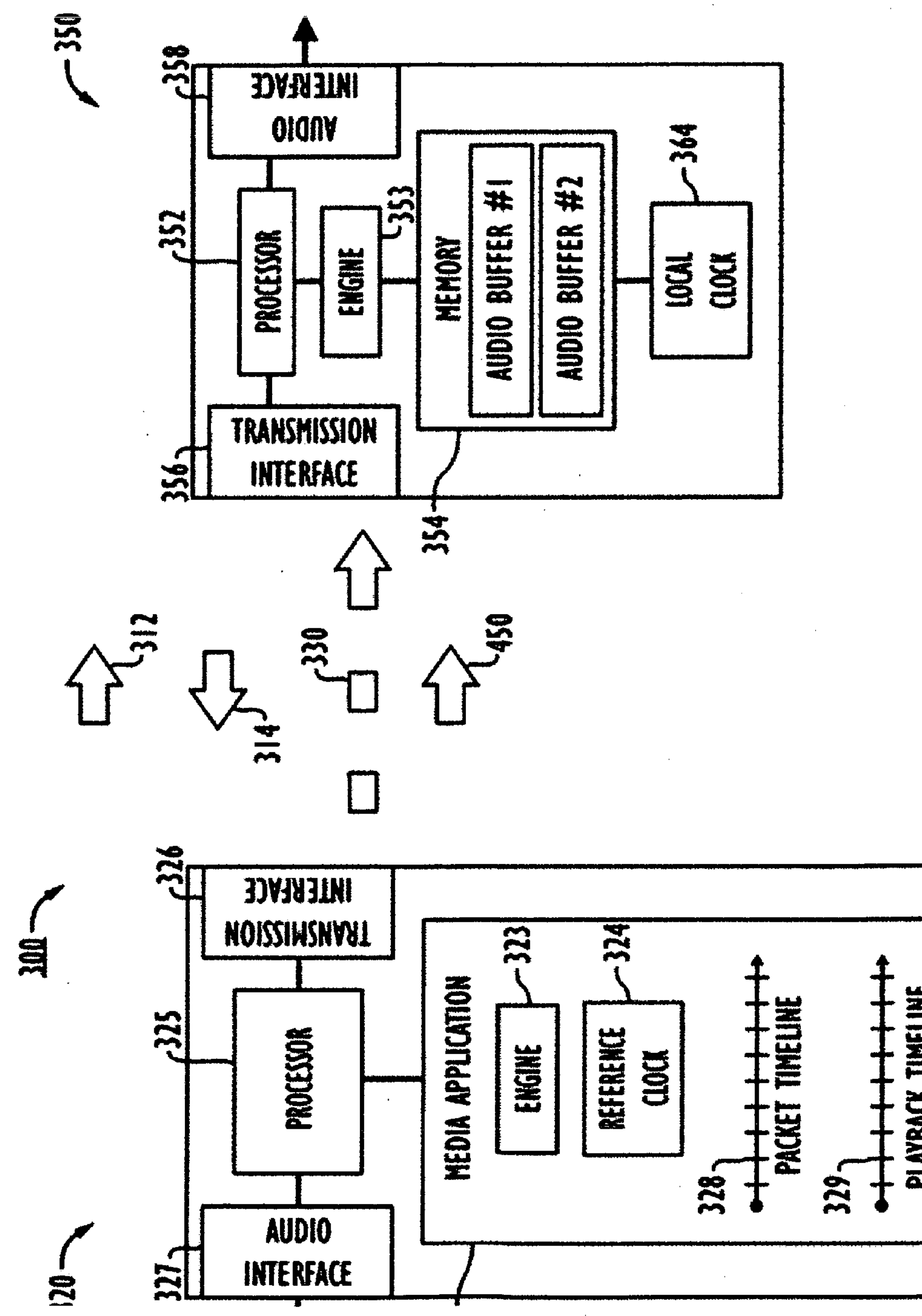

350
358
AUDIO INTERFACE
352
PROCESSOR
ENGINE
353
MEMORY
AUDIO BUFFER #1
AUDIO BUFFER #2
364
LOCAL CLOCK
356
TRANSMISSION INTERFACE
354
312
314
330
450
300
120
326
TRANSMISSION INTERFACE
325
PROCESSOR
MEDIA APPLICATION
323
ENGINE
324
REFERENCE CLOCK
328
PACKET TIMELINE
329
PLAYBACK TIMELINE
327
AUDIO INTERFACE

FIG. 9

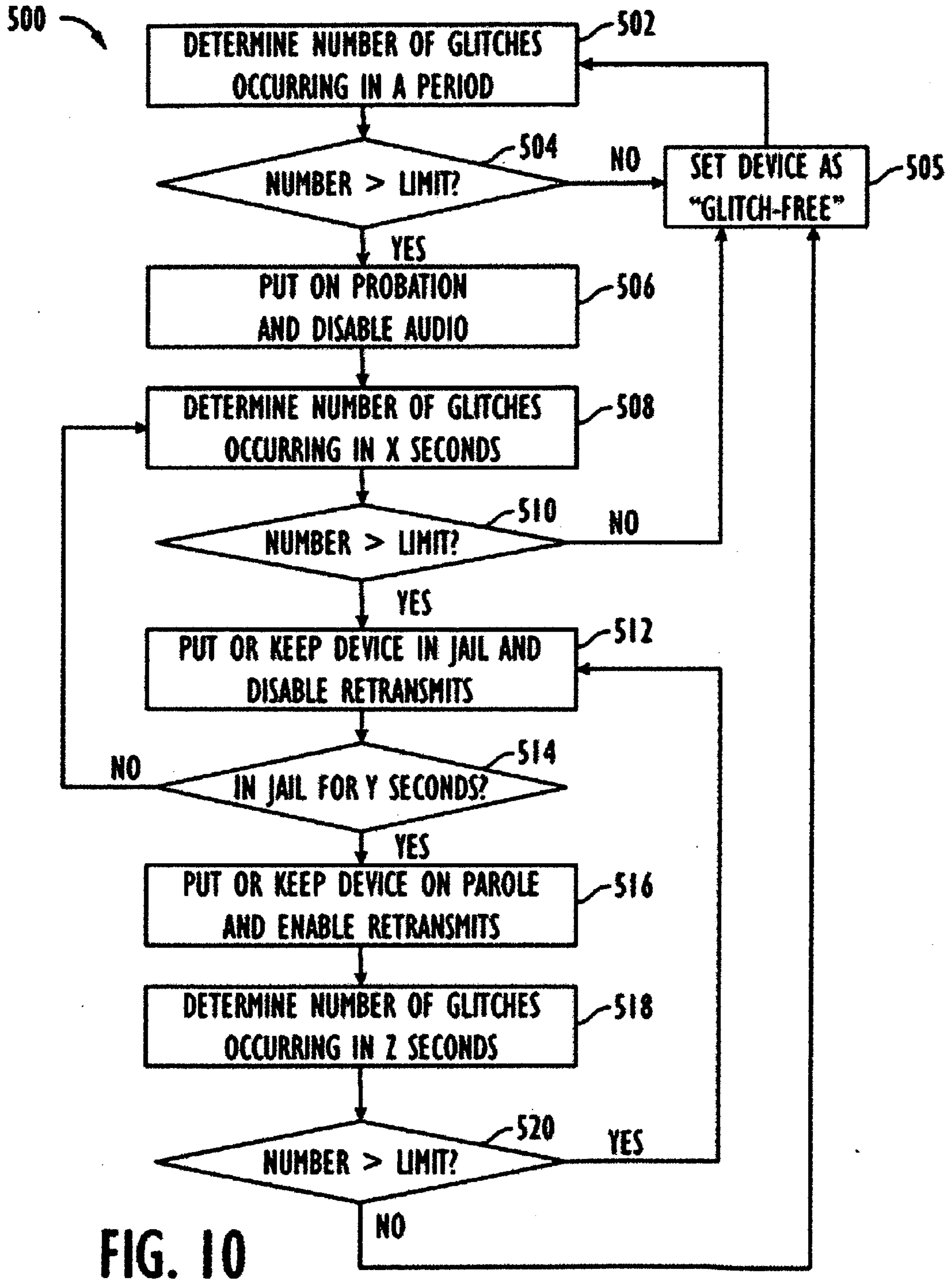

500
DETERMINE NUMBER OF GLITCHES OCCURRING IN A PERIOD
502
NUMBER > LIMIT?
504
NO
SET DEVICE AS "GLITCH-FREE"
505
YES
PUT ON PROBATION AND DISABLE AUDIO
506
DETERMINE NUMBER OF GLITCHES OCCURRING IN X SECONDS
508
NUMBER > LIMIT?
510
NO
YES
PUT OR KEEP DEVICE IN JAIL AND DISABLE RETRANSMITS
512
NO
IN JAIL FOR Y SECONDS?
514
YES
PUT OR KEEP DEVICE ON PAROLE AND ENABLE RETRANSMITS
516
DETERMINE NUMBER OF GLITCHES OCCURRING IN Z SECONDS
518
NUMBER > LIMIT?
520
YES
NO

FIG. 10

SYSTEM AND METHOD FOR SYNCHRONIZING MEDIA PRESENTATION AT MULTIPLE RECIPIENTS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application is a continuation-in-part of U.S. patent application Ser. No. 10/862,115, filed Jun. 4, 2004 and entitled "Networked Media Station," which is incorporated herein by reference in its entirety and to which priority is claimed.

FIELD OF THE DISCLOSURE

[0002] The subject matter of the present disclosure relates to a system and method for synchronizing presentation of media at multiple recipients or devices on a network.

BACKGROUND OF THE DISCLOSURE

[0003] With the increasing capacity and capability of personal computers, as well as improved multimedia interfaces for these computers, it has become popular to use personal computers as a repository for multimedia content, such as songs, movies, etc. Particularly with music, the increased popularity of storing multimedia information on a personal computer has resulted in a variety of products and services to serve this industry. For example, a variety of stand-alone players of encoded multimedia information have been developed, including, for example, the iPod, produced by Apple Computer of Cupertino, Calif. Additionally, services have been developed around these devices, which allow consumers to purchase music and other multimedia information in digital form suitable for storage and playback using personal computers, including, for example, the iTunes music service, also run by Apple Computer.

[0004] These products and services have resulted in an environment where many consumers use their personal computer as a primary vehicle for obtaining, storing, and accessing multimedia information. One drawback to such a system is that although the quality of multimedia playback systems for computers, e.g, displays, speakers, etc. have improved dramatically in the last several years, these systems still lag behind typical entertainment devices, e.g., stereos, televisions, projection systems, etc. in terms of performance, fidelity, and usability for the typical consumer.

[0005] Thus, it would be beneficial to provide a mechanism whereby a consumer could easily obtain, store, and access multimedia content using a personal computer, while also being able to listen, view, or otherwise access this content using conventional entertainment devices, such as stereo equipment, televisions, home theatre systems, etc. Because of the increasing use of personal computers and related peripherals in the home, it would also be advantageous to integrate such a mechanism with a home networking to provide an integrated electronic environment for the consumer.

[0006] In addition to these needs, there is also increasing interest in the field of home networking, which involves allowing disparate devices in the home or workplace to recognize each other and exchange data, perhaps under the control of some central hub. To date a number of solutions in this area have involved closed systems that required the purchase of disparate components from the same vendor. For example, audio speaker systems that allow computer-controlled switching of music from one location to another may be purchased as a system from a single vendor, but they may be expensive and/or may limit the consumer's ability to mix and match components of a home network from different vendors according to her own preferences. Thus, it would be beneficial to provide a mechanism by which various home networking components from differing vendors can nonetheless interact in a home network environment.

[0007] The subject matter of the present disclosure is directed to overcoming, or at least reducing the effects of, one or more of the problems set forth above.

SUMMARY OF THE DISCLOSURE

[0008] A system and method for delivering network media at multiple devices is disclosed. For example, the network media delivery system includes client devices and a host device. Each client device has a network interface for network communication, an engine for processing media data, and a media interface for delivering processed media. The host device, which can be a computer, establishes network communication links with the client devices, which can be networked media stations. The media data can be audio, video, or multimedia. In one embodiment, the network communication links are wireless links established between a wireless network interface on the host device and wireless network interfaces on the client devices.

[0009] The host device sends media data to the client devices via the network. The media data can be sent wirelessly as unicast streams of packets containing media data that are transmitted at intervals to each client device. In one embodiment, the host device controls processing of media data such that processed media is delivered in a synchronized manner at each of the client devices. In another embodiment, the host device controls processing of media data such that processed media is delivered in a synchronized manner at the host device and at least one client device.

[0010] The system uses Network Time Protocol (NTP) to initially synchronize local clocks at the client devices with a reference clock at the host device. The media data is preferably sent as Real-Time Transport Protocol (RTP) packets from the host device to the client device. The system includes mechanisms for periodic synchronization, stretching, and compressing of time at the local clocks to handle clock drift. In addition, the system includes mechanisms for retransmission of lost packets of media data. In one embodiment, the system can be used to deliver audio at multiple sets of speakers in an environment, such as a house, and can reduce effects of presenting the audio out of sync at the multiple sets of speakers to avoid user-perceivable echo.

[0011] The foregoing summary is not intended to summarize each potential embodiment or every aspect of the present disclosure.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] The foregoing summary, preferred embodiments, and other aspects of subject matter of the present disclosure will be best understood with reference to a detailed description of specific embodiments, which follows, when read in conjunction with the accompanying drawings, in which:

[0013] FIG. **1** illustrates an embodiment of a network media delivery system according to certain teachings of the present disclosure.

[0014] FIG. **2** illustrates an embodiment of a networked media station or client device.

[0015] FIG. **3** illustrates a process of operating the disclosed system in flowchart form.

[0016] FIG. **4** illustrates an embodiment of an interface of a media application operating on a host device of the disclosed system.

[0017] FIG. **5**A illustrates portion of the disclosed system having a host device delivering packets to multiple client devices.

[0018] FIG. **5**B illustrates portion of the disclosed system having a host device and client devices performing retransmission of lost packet information.

[0019] FIG. **6**A illustrates an embodiment of a packet requesting retransmission of lost packets.

[0020] FIG. **6**B illustrates an embodiment of a response to retransmission request.

[0021] FIG. **6**C illustrates an embodiment of a response to a futile retransmission request.

[0022] FIG. **7** illustrates portion of the disclosed system having a host device and multiple client devices exchanging time information.

[0023] FIG. **8**A illustrates an embodiment of a packet for synchronizing time.

[0024] FIG. **8**B illustrates an embodiment of a packet for announcing time.

[0025] FIG. **9** illustrates portion of the disclosed system having a host device and a client device.

[0026] FIG. **10** illustrates an algorithm to limit stuttering in playback of audio.

[0027] While the subject matter of the present disclosure is susceptible to various modifications and alternative forms, specific embodiments thereof have been shown by way of example in the drawings and are herein described in detail. The figures and written description are not intended to limit the scope of the inventive concepts in any manner. Rather, the figures and written description are provided to illustrate the inventive concepts to a person skilled in the art by reference to particular embodiments, as required by 35 U.S.C. § 112.

DETAILED DESCRIPTION

[0028] A network media delivery system having a host device and multiple client devices is described herein. The following embodiments disclosed herein are described in terms of devices and applications compatible with computer systems manufactured by Apple Computer, Inc. of Cupertino, Calif. The following embodiments are illustrative only and should not be considered limiting in any respect.

I. Components of the Network Media Delivery System

[0029] Referring to FIG. **1**, an embodiment of a network media delivery system **10** according to certain teachings of the present disclosure is illustrated. The system **10** includes a host device or computer system **20** and one or more networked media stations or client devices **50**, and various other devices. The system **10** in the present embodiment represents only one of several possible configurations and is meant to be illustrative only. Other possible configurations are discussed in the incorporated U.S. patent application Ser. No. 10/862,115. For example, the host device **20** can have a wired or wireless connection to each of the client devices **50** without the use of a hub or base station **30**, or the host device **20** can have a wireless connection to the hub or base station **30**. ¶The system **10** is used to distribute media (e.g., audio, video, multimedia, etc.) via network connections from the host device **20** to multiple client devices **50** located throughout an environment, such as a house, office, etc.

[0030] The host device **20** is a personal computer, such as an AirPort-equipped Mac or a Wi-Fi-compliant Windows-based PC. The client devices **50** are networked media stations, such as disclosed in incorporated U.S. patent application Ser. No. 10/862,115. The client devices **50** are plugged into wall sockets, which provide power to the client devices **50**, and are coupled to entertainment devices, such as amplifiers **80**, powered speakers, televisions, stereo systems, videocassette recorders, DVD players, home theatre systems, or other devices capable of delivering media known in the art.

[0031] An example of the client device **50** is discussed briefly with reference to FIG. **2**. The client device **50** includes an AC power adapter portion **52** and a network electronics portion **54**. The network electronics portion **54** includes a wired network interface **62**, a peripheral interface **64**, and a media interface **66**. As illustrated, the wired network interface **62** is an Ethernet interface, although other types of wired network interface known in the art could be provided. Similarly, the peripheral interface **64** is illustrated as a USB interface, although other types of peripheral interfaces, such as IEEE 1394 ("Firewire"), RS-232 (serial interface), IEEE 1284 (parallel interface), could also be used. Likewise, the media interface **66** is illustrated as an audio interface including both an analog lineout and an optical digital audio functionality. However, other media interfaces known in the art, such as a multimedia interface or a video interface using composite video, S-video, component video, Digital Video Interface (DVI), High Definition Multimedia Interface (HTMI), etc., could also be provided.

[0032] The network electronics portion **54** also includes a wireless networking interface **68**. The wireless network interface **68** preferably takes the form of a "Wi-Fi" interface according to the IEEE 802.11b or 802.11g standards know in the art. However, other wireless network standards could also be used, either in alternative to the identified standards or in addition to the identified standards. These other network standards can include the IEEE 802.11a standard or the Bluetooth standard, for example.

[0033] Returning to FIG. **1**, the host device **20** runs a media application **22**. In one exemplary embodiment, the media application **22** is iTunes software for media file management and playback produced by Apple Computer, Inc. In the present configuration, which is only one of several possibilities, the host device **20** is equipped with an Ethernet port that is connected via a cable **24** to a base station **30**. The base station **30** can be any variety of access points known in the art. Preferably, the base station **30** includes wireless access, routing, switching and firewall functionality. The base station **30** is connected via a cable **42** to a modem **40**, which receives an Internet connection through a connection **44**. Using this arrangement, multimedia files stored on host device **20** can be played using stereo amplifiers **80**, which are connected to client devices **50** using one of the audio interfaces on the client devices **50**. The host device **20** and the client devices **50** preferably communicate via a wireless network segment (illustrated schematically by connections **32**), but wired network segments formed by wired connections, such as Ethernet cables, could also provide communication between the host device and the client devices **50**. The client devices **50** communicate with the entertainment devices via a wired network segment **82**.

[**0034**] The client devices **50** act as wireless base stations for a wireless network and enable the host device **20** to deliver media (e.g., audio, video, and multimedia content) at multiple locations in an environment. For example, the client devices **50** are connected to stereo amplifiers **80** or other entertainment devices to playback media stored on the host device **20**. In one embodiment, a line level audio or a digital fiber optic type of connector connects the client devices **50** to the stereo amplifiers **80**. Either type of connector can plug into the multimedia port (**66**; FIG. **2**), which is a dual-purpose analog/optical digital audio mini-jack. To interface with stereo amplifiers **80**, a mini stereo to RCA adapter cable **82** is used, which connects to RCA-type right and left audio input ports on the stereo amplifier **80**. Alternatively, a Toslink digital fiber optic cable can be used, which would connect to digital audio input port on the stereo amplifiers **80**. These and other configurations are disclosed in incorporated U.S. patent application Ser. No. 10/862,115.

[**0035**] For the purposes of the present disclosure, the client devices **50** can also be connected to laptops **70** or personal computers that are capable of playing media (audio, video, etc.) so that the laptops and personal computers can also be considered entertainment devices. Moreover, the laptops **70** or personal computers can have the same functionality as both a client device **50** and an entertainment device so that the laptops **70** and personal computers can be considered both a client device and an entertainment device. Accordingly, the term "client device" as used herein is meant to encompass not only the networked media stations associated with reference numeral **50**, but the term "client device" as used herein is also intended to encompass any device (e.g., laptop, personal computer, etc.) compatible with the network media delivery system **10** according to the present disclosure. In the present disclosure, however, reference is made to client devices **50** for ease in discussion. Furthermore, the term "entertainment device" as used herein is meant to encompass not only stereo amplifiers **80** as shown in FIG. **1**, but the term "entertainment device" as used herein is also intended to encompass powered speakers, televisions, stereo systems, videocassette recorders, a DVD players, home theatre systems, laptops, personal computers, and other devices known in the art that capable of delivering media.

[**0036**] The client devices **50** receive media data from the host device **20** over network connections and output this media data to the entertainment devices. Although it is contemplated that audio, video, audio/video, and/or other forms of multimedia may be used, exemplary embodiments disclosed herein relate to sharing of audio with client devices **50** connected to entertainment devices, such as stereo amplifiers **80**, or with laptops **70** or other computers having internal speakers or the like. The audio can be stored on the host device **20** or can be obtained from the Internet **46**. However, it will be appreciated that the teachings of the present disclosure can be applied to video, audio/video, and/or other forms of multimedia in addition to the audio in the exemplary embodiments disclosed herein. Furthermore, in the discussion that follows, various details of the network media delivery system are implemented using hardware and software developed by Apple Computer, Inc. Although certain details are somewhat specific to such an implementation, various principles described are also generally applicable to other forms of hardware and/or software.

[**0037**] During operation, the system **10** delivers the same audio in separate locations of an environment (e.g., multiple rooms of a home). The system **10** addresses several issues related to playing the same audio in multiple, separate locations. One issue involves playing the audio in the separate locations in a synchronized manner with each other. Because the host device **20** and the client devices **50** have their own processors, memory, and transmission interfaces, sending or streaming audio from the host device **20** to the client devices **50** through a wireless or wired communication link will not likely result in synchronized playing of the audio at the separate locations. In addition, the client device **50** may be connected to different types of entertainment devices, which may have different latency and playback characteristics. It is undesirable to play the same audio in the separate locations out of sync because the listener will hear echoes and other undesirable audio effects. The system **10** addresses this issue by substantially synchronizing the playing of the audio in each location so that echo and other effects can be avoided. It should be noted that the level of precision required to substantially synchronize the playing of media at each location depends on the type of media being played, the perceptions of the user, spatial factors, and other details specific to an implementation.

[**0038**] Another issue related to playing of the same audio involves how to handle lost audio data at the separate locations. To address this issue, the disclosed system **10** preferably uses a retransmission scheme to recover lost audio. These and other issues and additional details of the disclosed network media delivery system are discussed below.

II. Process of Operating the System

[**0039**] Referring to FIG. **3**A, a process **100** of operating the network media delivery system of the present disclosure is illustrated in flowchart form. During discussion of the process **100**, reference is concurrently made to components of FIG. **1** to aid understanding. As an initial step in the process **100**, network discovery is performed, and the networked client devices **50** and other configured devices (e.g., a configured laptop **70**) publish or announce their presence on the network using a predefined service type of a transfer control protocol (Block **102**). The host device **20** browses the local sub-net for the designated service type (Block **104**).

[0040] The network discovery is used to initiate the interface between the host device **20** and client devices **50** and other compatible devices over the network of the system **10**. One example of such a network discovery uses Bonjour, which is a technology that enables automatic discovery of computers, devices, and services on IP networks. Bonjour uses standard IP protocols to allow devices to find each other automatically without the need for a user to enter IP addresses or configure DNS servers. Various aspects of Bonjour are generally known to those skilled in the art, and are disclosed in the technology brief entitled "MAC OS X: Bonjour," dated April 2005, and published by Apple Computer, which is incorporated herein by reference in its entirety. To provide the media sharing functionality between the host device **20** and the client devices **50**, the client devices **50** advertise over the network that they support audio streaming and particular audio capabilities (e.g., 44.1 kHz sample rate, 16-bit sample size, and 2-channel/stereo samples). The client devices **50** may also advertise security, encryption, compression, and other capabilities and/or parameters that are necessary for communicating with the client devices **50**.

[0041] When complaint client devices **50** are discovered, the addresses and port numbers of the discovered devices **50** are stored for use by the system **10**. Then, the media application **22** displays information about the found client devices **50** in a user interface operating on the host device **20** (Block **106**). In one embodiment, for example, the media application **22** discovers the client devices by obtaining information of the user's step up of computers and networks for their house, office, or the like from another application containing such information. In another embodiment, for example, the media application **22** discovers the client devices **50** and recognizes these client devices **50** as potential destinations for audio data. Then, the media application **22** automatically provides these recognized devices **50** as part of a selectable destination for audio playback in a user interface.

[0042] FIG. **4** shows an example of a user interface **200** associated with the media application, such as iTunes. Among other elements, the user interface **200** shows an icon **202** for selecting playback locations (e.g., networked client devices and other playback devices located in a house), which have detected on the network. A user may select the icon **202** to access a pop-up menu **204** in which the user can activate/deactivate (i.e., check or uncheck) one or more of the playback locations as destinations for audio playback. Of course, the user interface **200** can display possible destinations for audio playback in a number of ways. For example, the display of possible destination can include a network schematic of the user's dwelling, office, or the like, that shows possible destination, or the display can be customized by the user.

[0043] Returning to FIG. **3**A, the user selects one or more of the client devices to be used for playback in the user interface (Block **108**). The host device **20** then uses Real-Time Streaming Protocol (RTSP) to set up and control the audio stream, and the host device **20** initiates an RTSP connection to each of the selected client devices **50** to determine which set of features the devices **50** support and to authenticate the user (if a password is required) (Block **110**). On the host device **20**, the user can then start playback using the user interface of the media application **22** (Block **112**). The host device **20** makes an RTSP connection to each client device **50** to set it up for playback and to start sending the audio stream (Block **114**). The host device **20** then sends a command to each client device **50** to initiate playback (Block **116**). When each client device **50** receives the command, the device **50** negotiates timing information via User Datagram Protocol (UDP) packet exchanges with the host device **20** (Block **118**). Each client device **50** then determines whether the timing negotiation either succeeds or fails (Block **119**). The client devices **50** do not respond to the command to initiate playback until the timing negotiation either succeeds or fails. The timing negotiation occurs early to guarantee that the client devices **50** have the initial timing information needed to synchronize their clocks with the host device **20** before any audio packets are processed by the client devices **50**. ¶If the negotiation succeeds, the client device **50** can be used for playback (Block **120**). If the negotiation fails, however, the associated client device **50** can perform a number of possible operations (Block **121**). For example, the client device **50** can return an error to the host device **20** in response to the command, and the session on this device **50** can be terminated. In another possible operation, the associated client device **50** can retry to negotiate the timing information. Alternatively, the associated client device **50** can ignore the fact that negotiating timing information has failed. This may be suitable when the user is not interested in the audio playing in synchronized manner in the multiple locations associated with the client devices **50**. For example, the client device may be located by the pool or out in the garage and does not necessarily need to deliver the audio in synch with the other devices.

[0044] During playback at Block **120**, the host device **20** sends audio data to the client devices **50**, which process the audio data and deliver processed audio to the connected entertainment devices. An example of the process of playing back audio is discussed below with reference to the flowchart of FIG. **3**B with concurrent reference to element numerals of FIG. **1**. Various buffering, error checking, and other data transfer steps have been omitted from the general description of FIG. **3**B.

[0045] As discussed above, the host device **20** is connected to a wireless network established by the access point **30**, which can also provide for a shared connection to the Internet or other network **46**. The client devices **50** are also connected to the wireless network and have their multimedia ports connected to stereo amplifiers **80** or other entertainment device having output speakers or other multimedia output capability. A digital media file (e.g., a song in ACC format) is stored on the host device **20**. Once playback is started (Block **122**), the host device **20** transcodes a portion of the media file from the format (e.g., AAC) in which it is stored to a format that is understood by client device **50** (Block **124**). This transcoding step is not necessarily required if the file is stored on the host device **20** in a format that is understood by the client device **50**. In any case, a block of audio data for transmission is created (Block **126**). This audio data is preferably compressed and encrypted (Block **128**). Encryption is not necessarily required, but it is advantageous for digital rights management purposes.

[0046] The host device **20** then transmits the audio data over the wireless network to the client devices **50** (Block **130**). The client devices **50** decrypt and decompress the received audio data (Block **132**), and the client devices **50** decode the audio data based on the encoding performed in Block **124** (Block **134**). The decoding results in raw audio data, which may be, for example, in the form of PCM data. This data is converted to analog audio signals by digital-to-audio converters (DAC) (Block **136**), and the audio signals are output to the stereo amplifiers **80** for playing with their loudspeakers (Block **138**).

[**0047**] With the benefit of the description of the components of the disclosed network media delivery system and its process of operation provided in FIGS. **1** through **4**, the discussion now turns to details related to how data is transferred between the host device and client devices, how lost data is handled, and how playback is synchronized, in addition to other details disclosed herein.

III. Network Transport Used for the System

[**0048**] To transfer audio data and other information, the network media delivery system **10** of the present disclosure preferably uses User Datagram Protocol (UDP) as its underlying transport for media data. UDP is beneficial for synchronized playback to the multiple client devices **50** because synchronized playback places time constraints on the network protocol. Because audio is extremely time sensitive and has a definite lifetime of usefulness, for example, a packet of media data, such as audio, can become useless if it is received after a point in time when it should have been presented. Accordingly, UDP is preferred because it provides more flexibility with respect to the time sensitive nature of audio data and other media data.

[**0049**] To use UDP or some similar protocol, the disclosed system is preferably configured to handle at least a small percentage of lost packets. The lost packets can be recovered using Forward Error Correction (FEC), can be hidden using loss concealment techniques (e.g. repetition, waveform substitution, etc.), or can be recovered via retransmission techniques, such as those disclosed herein. Although UDP is preferred for the reasons set forth herein, Transmission Control Protocol (TCP) can be used. Depending on the implementation, retransmission using TCP may need to address problems with blocking of transmissions. If a TCP segment is lost and a subsequent TCP segment arrives out of order, for example, it is possible that the subsequent segment is held off until the first segment is retransmitted and arrives at the receiver. This can result in a chain reaction and effective audio loss because data that has arrived successfully and in time for playback may not be delivered until it is too late. Due to some of the retransmission difficulties associated with TCP, the Partial Reliability extension of Stream Control Transmission Protocol (SCTP) can provide the retransmission functionality. Details related to the Partial Reliability of SCTP are disclosed in RFC 3758, which can be obtained from http://www.ieff.org/rfc/rfc3758.txt, which is incorporated herein by reference.

[**0050**] UDP is preferred for time critical portions of the protocol because it can avoid some of the problems associated with blockage of transmission. For example, UDP allows the host's media application **22** to control retransmission of lost data because the media application **22** can track time constraints associated with pieces of audio data to be delivered. Based on the known time constraints, the media application **22** can then decide whether retransmission of lost packets of audio data would be beneficial or futile. All the same, in other embodiments, time critical portions of the disclosed system, such as time syncing, can be implemented using UDP, and audio data delivery can use TCP with a buffering system that addresses blocking problems associated with TCP.

IV. Audio Streaming and Playback with System

[**0051**] Before discussing how the client devices negotiate timing information in order to play audio in synchronization, the discussion first addresses how the disclosed system streams audio for playback. Referring to FIG. **5**A, a portion of the disclosed system **300** is shown with a host device **320** and at least two client devices **350**A-B. Each of the client devices **350** has a processor **352**, a memory **354**, a transmission interface **356**, and an audio interface **358**. The client devices **350** also include a UDP stack and can include a TCP stack depending on the implementation. As noted previously with reference to the client device of FIG. **2**, the transmission interfaces **356** can be a Wi-Fi-compatible wireless network interface, and the audio interface **358** can provide an analog and/or an optical digital output. The processor **352** and memory **354** can be conventional hardware components known in the art. The memory **354** has two audio buffers **361** and **362**. Although not shown in FIG. **5**A, each of the client devices **350** has a local clock, a playback engine, and other features.

[**0052**] The host device **320** uses several commands to set up a connection with and to control operation of the client devices **350**. These commands include ANNOUNCE (used for identification of active client devices), SETUP (used to setup connection and operation), RECORD (used to initiate playback at client devices), PAUSE (used to pause playback), FLUSH (used to flush memory at the client devices), TEARDOWN (used to stop playback), OPTIONS (used to configure options), GET_PARAMETER (used to get parameters from the client devices), and SET_PARAMETER (used to set parameters at the client devices).

[**0053**] Preferably, the client devices **350** are authenticated when initially establishing a connection to the media application **322** running on the host device **320**. Upon successful authentication, the media application **322** opens network connections to the transmission interface **356** of the client devices **350**. Preferably, network connections between the host device **320** and the client devices **350** are separated into an audio channel for sending audio data and a control channel used to set up connection and operation between the devices **320** and **350**. However, a single channel could be used for data and control information. Once the connections are established, the host device **320** begins sending data to the client devices **350**. In turn, the client devices **350** receive the audio data, buffer some portion of the data, and begin playing back the audio data once the buffer has reached a predetermined capacity.

[**0054**] Communication between the host device **320** and the client devices **350** preferably uses the Real Time Streaming Protocol (RTSP) standard. The media application **322** at the host device **320** preferably uses Real-Time Transport Protocol (RTP) encapsulated in User Datagram Protocol (UDP) packets **330** to deliver audio data from the host device **320** to the client devices **350**. RTSP, RTP, and UDP are standards known to those skilled in the art. Therefore, some implementation details are not discussed here. Details of RTSP can be found in "Real-Time Streaming Protocol," RFC 2326, which is available from http://www.ietf.org/rfc/

rfc2326.txt and which is hereby incorporated by reference in its entirety. Details of RTP can be found in "Real-Time Transport Protocol," RFC 3550, which is available from http://www.ietf.org/rfc/rfc3550.txt and which is hereby incorporated by reference in its entirety.

[0055] The packets **330** have RTP headers and include both sequence numbers and timestamps. The data payload of the RTP packets **330** contains the audio data to be played back by the client devices **350**. The media files, from which the packets **330** are derived, can be stored on host device **320** in one or more formats, including, for example, MP3 (Motion Picture Expert's Group Layer 3), AAC (Advanced Audio Coding a/k/a MPEG-4 audio), WMA (Windows Media Audio), etc. Preferably, the media application **322** running on the host device **320** decodes these various audio formats to construct the packets **330** so that the client devices **350** do not need decoders for multiple formats. This also reduces the hardware performance requirements of the client devices **350**. Another advantage of performing decoding on the host device **320** is that various effects may be applied to the audio stream, for example, cross fading between tracks, volume control, equalization, and/or other audio effects. Many of these effects would be difficult or impossible to apply if the client device **350** were to apply them, for example, because of the computational resources required. Although not preferred in the present embodiment, other embodiments of the present disclosure can allow for decoding at the client devices **350** for audio and other forms of media.

[0056] The host device **320** preferably uses a separate unicast stream **310**A-B of RTP packets **330** for each of the client devices **350**A-B. In the present embodiment, the separate unicast streams **310**A-B are intended to deliver the same media information (e.g., audio) to each of the client devices **350**A-B so that the same media can be presented at the same time from multiple client devices **350**A-B. In another embodiment, each of the separate unicast streams **310**A-B can be used to deliver separate media information (e.g., audio) to each of the client devices **350**A-B. The user may wish to unicast separate media information in some situations, for example, if a first destination of a first unicast stream of audio is a client device in a game room of a house and a second destination of a second unicast stream of different audio is a client device in the garage of the house. Therefore, it may be preferred in some situations to enable to the user to not only select sending the same media information by unicast streams to multiple client devices by to also allow the user to send different media information by separate unicast streams to multiple client devices. The user interface **200** of FIG. **4** can include a drop down menu or other way for the user to make such a related selection.

[0057] Separate unicast streams **310** are preferred because multicasting over wireless networks can produce high loss rates and can be generally unreliable. All the same, the disclosed system **300** can use multicasting over the wireless network. In general, though, bandwidth limitations (i.e. fixed multicast rate), negative effects on unicast performance (low-rate multicast slows down other unicast traffic due to multicast packets taking longer), and loss characteristics associated with multicasting over wireless (multicast packets are not acknowledged at the wireless layer) make multicasting less desirable than using multiple, unicast streams **310**A-B as preferred. Use of multiple, unicast streams **310**A-B does correspond to an increase in bandwidth as additional client devices **350** are added to a group of designated locations for playback. If the average compression rate for audio data is about 75%, the increase in bandwidth associated with multiple, unicast streams **310**A-B may correspond to about 1 Mbit/sec bandwidth required for each client device **350** so that the host device **320** can send compressed audio data to the access point (e.g., **30**; FIG. **1**) and another 1 Mbit/sec so that the access point can forward the compressed audio data to the client device **350**.

[0058] Once an RTSP session has been started and the RECORD command has been sent from the host device **320** to the client devices **350**, the host device **320** begins sending normal RTP packets **330** containing the audio data for playback. These RTP packets **330** are sent at regular intervals, based on the number of samples per second, which can be about 44,100 Hz for audio. The RTP packets **330** are sent at the regular intervals in a throttled and evenly spaced manner in order to approximate the audio playback rate of the remote client devices **350** because the UDP-based connection does not automatically control the sending of data in relation to the rate at which that data is consumed on the remote client devices **350**.

[0059] Because each of the multiple client devices **350** has their own audio buffers **361**, **362**, network conditions, etc., it may not be desirable to use a feedback scheme when sending the packets **330**. Accordingly, the host device **320** sends audio data at a rate that preferably does not significantly under-run or over-run a playback engine **353** of any of the remote client devices **350**. To accomplish this, the host device **320** estimates a fixed delay **340** to insert between packets **330** to maintain the desired audio playback rate. In one embodiment, the packets **330** of audio data are sent with a delay of about 7.982-ms between packets **330** (i.e., 352 samples per packet/44,100 Hz=~7.982-ms per packet), which corresponds to a rate of about 125 packets/sec. Because the delay **340** is fixed, each of the client devices **350** can also detect any skew between its clock and the clock of the sending host device **320**. Then, based on the detected skew, each client device **350** can insert simulated audio samples or remove audio samples in the audio it plays back in order to compensate for that skew.

[0060] As alluded to above, the RTP packets **330** have timestamps and sequence numbers. When an RTP packet **330** is received by a client device **350**, the client device **350** decrypts and decompresses the payload (see Encryption and Compression section below), then inserts the packet **320**, sorted by its timestamp, into a packet queue. The two audio buffers **361** and **362** are alternatingly cycled as audio is played back. Each audio buffer **361** and **362** can store a 250-ms interval of audio. The received RTP packets in the packet queue are processed when one of the two, cycling audio buffers **361** and **362** completes playback. In one embodiment, the audio is USB-based so this is a USB buffer completion process.

[0061] To process the queued packets, the engine **353** assembles the queued RTP packets in one of the audio buffers **361** or **362**. During the assembly, the engine **353** calculates when each of queued RTP packets should be inserted into the audio stream. The RTP timestamp in the packets combined with time sync information (see the Time Synchronization section below) is used to determine when to insert the packets. The engine **353** performs this assembly process and runs through the queued packets to fill the inactive audio buffer **361** or **362** before the currently playing audio buffer **361** or **362** has completed. Because each of the audio buffers **361** and **362** can store 250-ms of audio, the client device **350** has a little less than 250-ms to assemble all the RTP packets, conceal any losses, and compensate for any clock skew. If there are any gaps in the audio (e.g., the device's audio clock is skewed from the host's audio clock, a packet was lost and not recovered, etc.), then those gaps can be concealed by inserting simulated audio samples or removing existing audio samples.

V. Encryption and Compression

[**0062**] For digital rights management purposes, it is desirable to determine whether the client devices **350** are authorized to receive an audio data stream and/or whether the communications links between the host device **320** and the client devices **350** are secure (encrypted). This requires some form of authentication, which is preferably based on a public key/private key system. In one embodiment, each client station **350** is provided with a plurality of private keys embedded in read only memory (ROM). The media application at the host device **320** is then provided with a corresponding plurality of public keys. This allows identification data transmitted from the networked client devices **350** to the media application to be digitally signed by the client device **350** using its private key, by which it can be authenticated by the media application at the host device **320** using the appropriate public key. Similarly, data sent from the media application at the host device **320** to the networked client stations **350** is encrypted using a public key so that only a client device **350** using the corresponding private key can decrypt the data. The media software and networked media station can determine which of their respective pluralities of keys to use based on the exchange of a key index, telling them which of their respective keys to use without the necessity of transmitting entire keys.

[**0063**] In addition to encryption, the decoded audio data is preferably compressed by host device **320** before transmission to the client devices **350**. This compression is most preferably accomplished using a lossless compression algorithm to provide maximum audio fidelity. One suitable compressor is the Apple Lossless Encoder, which is available in conjunction with Apple's iTunes software. The client devices **350** require a decoder for the compression codec used.

[**0064**] The RTP packets **330** are preferably compressed using the Apple Lossless algorithm and are preferably encrypted using the Advanced Encryption Standard (AES) with a 128-bit key size. Loss is still inevitable even though the system **300** uses a UDP-based protocol that attempts to recover from packet loss via retransmission and/or Forward Error Correction (FEC). For this reason, encryption and compression preferably operate on a per-packet basis. In this way, each packet **330** can be completely decoded entirely on its own, without the need for any surrounding packets **330**. The Apple Lossless algorithm is used to compress each individual packet **330** rather than compressing a larger stream of audio and packetizing the compressed stream. Although compressing each individual packet **330** may reduce the effectiveness of the compression algorithm, the methodology simplifies operation for the client devices **350** and allows them to be more tolerant to packet loss. Although compression rates are highly dependent on the content, music audio can have an average compression rate of about 75% of the original size when used by the disclosed system **300**.

[**0065**] The AES-128 algorithm is used in frame-based cipher block chaining (CBC) mode to encrypt payloads of the RTP packets **330** and the RTP payload portion of RTCP retransmission packets (**380**; FIG. **5**B) discussed below. Because each packet **330** represents a single audio frame, no other packets are required to decrypt each packet correctly. The system preferably supports any combination of encryption and compression, such as both encryption and compression, encryption only, compression only, or neither encryption nor compression. Encryption and compression are configured during the RTSP ANNOUNCE command. The format used to configure encryption and compression is based on the Session Description Protocol (SDP) and embedded as RTSP header fields. Compression uses an SDP "m" (media description) combined with an "rtpmap" and "fmtp" to specify the media formats being used numerically and how those numbers map to actual compression formats and algorithms.

VI. Retransmission of Lost Packets of Audio Data

[**0066**] As noted above, the RTP packets **330** received from the host device **320** have RTP sequence numbers. Based on those RTP sequence numbers, the client device **350** can determine whether packets **330** that have been lost during transmission or for other reasons. The lost RTP packets **330** cannot be queued for playback in the audio buffers **361** and **362** of the client devices **350** so that gaps will result in the audio. To address this issue, the client devices **350** requests that the lost packet(s) be retransmitted. Referring to FIG. **5**B, portion of the disclosed system **300** is shown again to discuss how the system **300** attempts to retransmit packets lost during original transmission.

[**0067**] To handle retransmissions, the system **300** preferably uses Real-Time Transport Control Protocol (RTCP) when packet loss is detected. As note above, the sequence numbers associated with the received RTP packets (**330**; FIG. **5**A) are used to determine if any packets have been lost in the transmission. If there is a gap in the sequence numbers, the client device **350** sends a retransmission request **370** to the sender (e.g., host device **320** or other linked client device **350**) requesting all the missing packets. In one embodiment, the retransmission request **370** can request up to a maximum of 128 lost packets per detected gap.

[**0068**] In response to the retransmission request **370**, the host device **320** sends one or more retransmission responses **380** for lost packets. Due to limitations of the maximum transmission unit (MTU) on RTCP packet sizes, only one response can be sent per retransmission response packet **380**. This means that a single retransmission request packet **370** from a device **350** may generate up to 128 retransmission response packets **380** from the host device **320** if all of the lost packets are found in the host's recently sent packets.

[**0069**] Because RTP does not currently define a standard packet to be used for retransmissions, an RTP extension for an RTCP Retransmission Request packet is preferably defined. FIG. **6**A shows an example of an RTCP Retransmit

Request Packet **370** for use with the disclosed system. The Sequence Number Base refers to the sequence number of the first (lost) packet requested by this RTCP Retransmit Request Packet **370**. The Sequence Number Count refers to the number of (lost) packets to retransmit, starting at the base indicated.

[0070] In FIG. **5**A, the client device **350** sending the RTCP Retransmission Request packet **370** tracks the retransmission requests that it sends in a queue to facilitate sending additional requests if a response to the retransmission request **370** is not received in a timely manner. When a retransmission request **370** has not been responded to in a timely manner, another retransmission request **370** is sent from the client device **350**. The process of retrying can be continued until a maximum time has elapsed since the first retransmission request **370** was sent. After that maximum time, it is likely too late to deal with the lost packet anyway because the lost packets time for insertion in one of the audio buffers **361** or **362** has passed.

[0071] When multiple, contiguous packets have been lost, the initial retransmit request **370** includes all the missing packets. However, if a response **380** is not received in a timely manner, the missing packets are spread out among multiple requests **370** over time when reattempts are made. Spreading out among multiple requests can maintain a uniform delivery of request and response packets. This also prioritizes packets by time and defers delivery of packets whose presentation time is later.

[0072] When the host device **320** receives a retransmission request **370**, the host device **320** searches a list of recently sent packets stored at the device **320**. If the requested packet in the request **370** is found, the host device **320** sends a retransmission response **380** to the client device **350**. An example of an RTP extension for an RTCP Retransmit Response Packet **380** is shown in FIG. **6**B. The RTCP Retransmit Response Packet **380** includes the complete RTP packet (e.g., header and payload) being retransmitted. The retransmission packet **380**, however, is only sent to the sender of the retransmission request **370**, unlike the normal RTP packets (**330**; FIG. **5**A) that are sent to all devices participating in the session.

[0073] If the requested packet is not found by the host device **320**, however, a negative response **390** is sent so the corresponding client device **350** knows that any further attempt to request that particular packet is futile. An example of an RTP extension for an RTCP Futile Retransmit Response Packet **390** is shown in FIG. **6**C. The RTCP Futile Retransmit Response Packet **390** includes the 16-bit sequence number of the failed packet followed by a 16-bit pad containing zero.

[0074] In FIG. **5**B, the client device **350** receiving a retransmission response packet **380** inserts the packet **380** into the packet queue in the same way used for inserting packets received as part of the normal RTP packet stream discussed above with reference to FIG. **5**A. By definition, however, the retransmission response packet **380** is already out-of-sequence and, therefore, does not trigger new retransmission requests based on its sequence number. If an existing packet already exists at the same timestamp as the incoming packet, either via the normal RTP stream or via retransmission, the packet is dropped as a duplicate.

[0075] Scheduling retransmission is based on regular reception of RTP packets (**330**; FIG. **5**A) rather than explicit timers. This simplifies the code required and reduces retransmission overhead, but it also throttles retransmission during burst outages (e.g. wireless interference resulting in packet loss during a period). Since retransmissions only occur when RTP packets **330** are received, retransmissions are deferred beyond a possible window when packets **330** may have been lost anyway.

VII. Controlling Relative Volume at Multiple Client Devices During Playback

[0076] Because the disclosed system **330** plays music at multiple locations at the same time, it may be desirable to be able to adjust the volume at each location individually. The disclosed system **300** supports individual volume control by using a relative volume setting specified using a header field as part of an RTSP SET_PARAMETER request. The volume is expressed as a floating-point decibel level (e.g. 0 dB for full volume). In addition to volume, the disclosed system **330** can set other parameters related to the delivery of media at multiple locations using similar techniques. For example, the disclosed system **300** can be used to set equalization levels at each location individually.

VII. Time Synchronization Between Host Device and Multiple Client Devices

[0077] Referring to FIG. **7**, portion **300** of the disclosed system is shown having a host device **320** and multiple client devices **350** exchanging timing information. To play the same audio on the multiple client devices **350** in synchronization with each other, the timebase on the multiple client devices **350** is synchronized with a reference clock **324** on the host device **320**. As noted previously, the host device **320** can be a Mac or Windows-based system running the media application **322**. The host device **320** does not need to run any special server software, and only the media application **322** according to the present disclosure is required. The reference clock **324** at the host device **320** does not need to be synchronized with an external clock, such provided by an NTP server. Rather, the client devices **350** only need to be synchronized to the same reference clock **324** even if that clock **324** is wrong with respect to an external clock.

[0078] The reference clock **324** is maintained within the media application **322** running on the host device **320**. If the host device **320** is a Macintosh computer, then the reference clock **324** can use the PowerPC timebase registers. If the host device **320** is a Windows-based computer, the reference clock **324** can use the Pentium performance counter registers. The reference clock **324** of the host's media application **322** is separate from the normal wall-clock time of the host device **320**, which is maintained by an NTP agent and synchronized to an external clock. The reference clock **324** of the host's media application **322** does not need to be synchronized to an external clock and in some cases this would actually be undesirable. For example, a time difference between the reference clock **324** and the local clock of a client device **350** can be explicitly skewed or adjusted to account for spatial effects or differences, such at the client device **350** being located farther away than another. In addition, there may be situations where a user may want to intentionally skew the clocks to produce effects. Accordingly, the user interface associated with the disclosed system **300**, such as interface **200** of FIG. **4**, may include a drop-down menu or other control for intentionally manipulating skew.

[0079] To synchronize the timebase between the client devices **350** and the host device **320**, the media application **322** uses time sync information based on the principals of the Network Time Protocol (NTP) encapsulated in Real-Time Transport Control Protocol (RTCP) packets. Preferably, NTP is not used directly to avoid collisions with existing NTP services (e.g., date/time synchronization with an external clock) and to avoid permission issues due to NTP's use of a privileged port number. Even though the time sync information of the media application **322** is encapsulated in RTCP packets, the time synchronization works substantially the same as NTP and will be referred to as NTP henceforth. NTP is known in the art and provides the basis for inter-media synchronization support in the Real-Time Transport Protocol (RTP). Details of NTP can be found in "Network Time Protocol," RFC 1305, which is available from http://www.ietf.org/rfc/rfc1305.txt and is incorporated herein by reference in its entirety.

[0080] Techniques of NTP, however, are preferably not used to provide moment-to-moment time directly to each client device **350** due to issues related to network latency, bandwidth consumption, and CPU resources. Accordingly, techniques of NTP are used for periodic synchronization of time. In addition, each client device **350** is provided with a high-resolution clock **364** based on the local clock hardware of each client device **350** (see Local Clock Implementation section below), the high-resolution clocks **364** are synchronized with the reference clock **324** of the host device **320** using the NTP techniques.

[0081] Synchronizing the local clocks **364** of the client devices **350** with the reference clock **324** preferably does not jump to a new time with every correction (referred to as stepping) because stepping can introduce discontinuities in time and can cause time to appear to go backward, which can create havoc on processing code that relies on time. Instead, the time synchronization techniques of the present disclosure preferably correct time smoothly using clock slewing so that time advances in a linear and monotonically increasing manner. In the clock slewing techniques of the present disclosure, frequent micro-corrections, below a tolerance threshold, are performed to the running clocks **364** at the client devices **350** to bring their timebase gradually in sync with the timebase of the reference clock **324** of the host's media application **322**. The clock slewing techniques also predict the relative clock skew between the local clocks **364** and the host's reference clock **324** by analyzing past history of clock offsets and disciplining the local clocks **364** to run at the same rate as the host's reference clock **324**.

[0082] Because a centralized reference clock **324** is used for several client devices **350** on a local network, one way to disseminate time information is to send broadcast/multicast NTP packets periodically from the host device **320** to the client devices **350**. Sending NTP packets by multicasting must account for losses and performance degradation that may result from the wireless 802.11b and 802.11g communication links between the host device **320** and the client devices **350**. Due to issues of performance degradation, loss rates, and lack of propagation delay information associated with broadcasting or multicasting, unicast NTP transactions **400** are preferably used.

[0083] As part of the unicast NTP transactions **400**, the client devices **350** periodically send unicast requests **410** to the host device **320** so that the client devices **350** can synchronize their clocks **364** with the reference clock **324**. Then, the client devices **350** use responses **420** from the host device **320** corresponding to their requests **410** to continually track the clock offset and propagation delay between the client device **350** and host device **320** so the client devices **350** can update their local clocks **364**. Thus, synchronization of the audio playback at the client devices **350** is achieved by maintaining local clocks **364** that are synchronized to the host device's clock **324**. Since all client devices participating in a particular session are synchronized to the reference clock **324**. When the clocks **324** and **364** are synchronized, the client devices **350** can play audio in-sync without ever communicating with each other.

[0084] With the timebase at the client devices **350** synchronized with the reference clock **324** at the host device **320**, the client devices **350** can use the synchronized timebase to determine when to playback packets of audio data. As noted previously, audio data is delivered to the client devices **350** using RTP packets (**330**; FIG. **5**A) that contain an RTP timestamp describing the time of a packet's audio relative to other packets in the audio stream. The client device **350** uses this timestamp information to reconstruct audio at the correct presentation time for playback. Accordingly, each client device **350** correlates the NTP timebase of its local clock **364** with the RTP timestamps provided in the RTP packets of the audio stream.

[0085] With respect to the unicast requests and responses **410** and **420** noted above, RTP does not define a standard packet format for synchronizing time. There is an RTCP sender report, which contains some timing information, but not everything that is needed to synchronize time (e.g., there is no originate time for receivers to determine the round trip time). There are also rules preventing sender reports from being sent before any RTP data has been sent, which is critical for playing the initial audio samples in sync.

[0086] Therefore, the host's media application **322** preferably defines an RTP extension for an RTCP TimeSync packet for the requests and responses **410** and **420**. An embodiment of an RTCP TimeSync packet **430** is shown in FIG. **8**A. The RTCP TimeSync Packet **430** includes a header, the RTP timestamp at NTP Transmit (T3) time, NTP Originate (T1) timestamp, most significant word; NTP Originate (T1) timestamp, least significant word; NTP Receive (T2) timestamp, most significant word; NTP Receive (T2) timestamp, least significant word; NTP Transmit (T3) timestamp, most significant word; NTP Transmit (T3) timestamp, least significant word. The Marker bit (M) is not used for these TimeSync packets **430**. The packet types (PT) include '210' for a client device request to synchronize time in a manner similar to an NTP client device request and include '211' for a host device response to a client device request. The 'RTP Timestamp' is the RTP timestamp at the same instant as the transmit time (T3). This should be 0. The times T1-T3 come from NTP and are used in the same manner as NTP.

[0087] In FIG. **7**, the RTCP TimeSync request packets **410** from the client devices **350** are sent once the RTSP RECORD command is received so that the client devices **350** can initially synchronize time. Then, the client devices **350** periodically send RTCP TimeSync request packets **410**. In one embodiment, the periodic intervals for synchronizing time can be at random intervals between two and three seconds apart. The RTCP TimeSync response packets **420** are sent by the host device **320** in response to receiving a valid RTCP TimeSync request packet **410**.

[0088] The host's media application **322** also defines an RTP extension for an RTCP TimeAnnounce packet **450**. The RTCP TimeAnnounce packets **450** are sent periodically (e.g., once a second) by the host device **320** to update the client devices **350** with the current timing relationship between NTP and RTP. The RTCP TimeAnnounce packets **450** can be sent sooner if the host device **320** changes the NTP to RTP timing relationship. For example, when a new song starts, the host's media application **322** can send a new RTCP TimeAnnounce packet **450** with the marker bit (M) set to indicate that the NTP to RTP timing relationship has changed.

[0089] As shown in the embodiment of FIG. **8**B, the RTCPTimeAnnounce Packet **450** includes an RTP timestamp; an NTP timestamp, high 32 bits; an NTP timestamp, low 32 bits; and an RTP timestamp when the new timeline should be applied. The Marker bit (M) is used to indicate an explicit change in the NTP to RTP timing relationship. The packet type (PT) is defined as '212' to indicate that the host device is announcing a new NTP to RTP relationship. The "RTP Timestamp" is the RTP timestamp at the same instant as the NTP timestamp. The "NTP Timestamp" is the NTP timestamp at the same instant as the RTP timestamp. The field "RTP Apply Timestamp" refers to the RTP timestamp when the new timeline should be applied.

IX. Local Clock Implementation at Host Device

[0090] Returning to FIG. **7**, the local clock **364** of the client device **350** is discussed in more detail. The local clock **364** maintains a 64-bit nanoseconds counter that starts at zero on system boot and uses the 60-Hz clock interrupt to increment the nanoseconds counter. When an interrupt occurs, the 32-bit timer counter is used to determine how much time has passed since the last clock interrupt. This determined amount of time since the last clock interrupt is referred to as the tick delta and is in units of 1/100 of a microsecond. The tick delta is then converted to nanoseconds and is added to the nanoseconds counter to maintain the current time. The tick delta is used in this manner to avoid drift due to interrupt latency.

[0091] To maintain more accurate time, it may be preferable to allow time to be adjusted gradually. Accordingly, the nanoseconds counter is adjusted in very small increments during each clock interrupt to "slew" to the target time. These small increments are chosen based on a fraction of the amount of adjustment needed and based on the tick delta. This prevents time from appearing to go backward so that time always increases in a linear and monotonic manner.

[0092] Additionally, the client device **350** can predict what the next NTP clock offset will be in the future to further adjust the local clock **364**. To make the prediction, the client device **350** uses a moving average of NTP clock offsets to estimate the slope of the clock skew between each of client device **350** and host device **320**. This slope is then extrapolated to estimate the amount of adjustment necessary to keep the local clock **364** at the client device **350** in sync with the reference clock **324**. The client device **350** then makes very small adjustments to the per-clock interrupt increment, in addition to the adjustments made for clock slewing, to simulate the faster or slower clock frequency of the host's reference clock **324**. This allows the local clock **364** to remain synchronized between NTP update intervals and may even allow the reference clock **324** to remain synchronized in the absence of future NTP clock updates).

X. Simulated Timelines for Audio Playback

[0093] Referring to FIG. **9**, additional details related to synchronized delivery of media with multiple client devices are discussed. In FIG. **9**, portion of the network media delivery system **300** is again illustrated. The host device **320** is schematically shown having the media application **322** and reference clock **324**, as described previously. In addition, the host device **320** is schematically shown having an engine **323**, a processor **325**, a transmission interface **326**, and an audio interface **327**. As disclosed herein, the host device **320** can be a computer. Therefore, the processor **325** can be a conventional computer processor, the transmission interface **326** can be a Wi-Fi compatible wireless network interface, and the audio interface **327** can be a sound card or the like for playing audio. In addition, the media application **322** can be a software program stored in memory on the computer and operating on the computer processor **325**. Furthermore, the media application **322** can include the engine **324** for processing media (e.g., audio) data and can include the reference clock **324** for synchronizing time.

[0094] To play audio in a synchronized manner on multiple client devices **350** (only one of which is shown in FIG. **9**), audio data needs to be scheduled for playback at a constant or consistent rate. One way to achieve this is for the media application **322** on the host device **320** to send packets **330** of audio data at a constant rate and to have the timeline for presenting that audio data with the client device **350** tied to the send rate of the packets **330**. For example, packets of audio data can be sent about every 7.982-ms (i.e., 352 samples per packet/44,100 Hz=~7.982-ms per packet, which corresponds to a rate of about 125 packets/sec), and the timeline for presenting that audio can correspond directly to this rate. While this works, the send rate of the packets **330** and the presentation timeline at the client device **350** must have a one-to-one correspondence, which can restrict the ability to buffer the audio data at the client device **350**. As discussed herein, buffering of the audio data at the client devices **350** is desirable for handling lost packets, clock skew, etc. If five seconds of buffering is desired at the client device **350**, there will be a five-second delay between the time when the audio data arrives at the client device and the time when it is actually played. Unfortunately, users can readily perceive such a high level of latency when buffering is used with such a one-to-one correspondence between the packet send rate and the presentation time of the audio.

[0095] To provide buffering without this high level of latency, the sending of packets **330** is preferably decoupled or separated from the timeline for presenting the audio data of those packets **330**. To achieve this, the media application **322** maintains two simulated timelines **328** and **329**. A first packet timeline **328** corresponds to when packets **330** should be sent, and a second playback timeline **329** corresponds to when the audio data in those packets **330** should be presented or delivered (i.e., played for the user). The separate timelines **328** and **329** allow the send rate of the packets **330** to vary as needed so that the system **300** can provide buffering without introducing latency. If more buffering is needed, for example, the packet send rate of the first packet timeline **328** can be temporarily increased to front-load the buffers in memory **354** on the client devices **350** and can be later reduced back to the real-time send rate of the packets **330**. The separate timelines **328** and **329** also avoid problems associated with fluctuations in the presentation time of audio caused by scheduled latency of the operating systems on the devices.

[**0096**] The second playback timeline **329**, which corresponds to when the audio data in the packets **330** should be presented or delivered, is constructed by the host device **320**. Using the reference clock **324** and a desired playback rate of the audio, the host device **320** estimates the number of audio samples that would have played at a given point in time at the client device **350** to construct the playback timeline **329**. This second playback timeline **329** is then published from the host device **320** to the client devices **350** as part of the time announcements **450** sent periodically from the host device **320** to the client devices **350**. As discussed in greater detail previously, the client device **350** uses the periodic time announcements **450** to establish and maintain the relationship between the RTP timestamps in the audio packets **330** and the corresponding NTP presentation time for the audio packets **330** so that the client device **350** can deliver the audio in synch with other devices.

[**0097**] By having the send rate of the packets **330** (represented by the packet timeline **328**) separate from the presentation time (represented by the playback timeline **329**), the periodic time announcements **450** are not designed to take effect immediately when received by the client devices **350** since the announcements **450** may come in advance of when they are effective. As noted previously, however, the time announcement packets **450** contain an additional RTP timestamp that indicates when the announced time should take effect at the client device **350**. Therefore, a time announcement packet **450** is saved at a client device **350** once it is received. When audio playback reaches the RTP timestamp of that saved time announcement packet **450**, the client device **350** applies the time change contained in that saved time announcement package **450**.

[**0098**] To play audio in a synchronized manner on multiple client devices **350** (only one of which is shown in FIG. **9**), it is also preferred to consider the amount of latency or delay between the time when the audio data is scheduled to be delivered at the device **350** and the time when the audio is actually delivered by the device **350** (and associated entertainment devices). Different types of client devices **350** (and associated entertainment devices) will typically have different latency characteristics. Accordingly, the disclosed system **300** preferably provides a way for each client device **350** to report its latency characteristics (and that of its associated entertainment device) to the host device **320** so that these latency characteristics can be taken into consideration when determining how to synchronize the playback of media at the client devices **350**.

[**0099**] Determination of the latency characteristics of the client devices **350** preferably occurs at initial set up of the system **300**. For example, the media application **322** at the host device **320** sends RTSP SETUP requests **312** to the client devices **350** at initial set up. In responses **314** to the RTSP SETUP requests **312**, the client devices **350** use a header field to report the latency characteristics associated with the client devices **350**. The values of the field are preferably given as the number of RTP timestamp units of latency. For example, a client device **350** having 250-ms of latency at a 44,100-Hz sample rate would report its audio-latency as 11025 RTP timestamp units. Based on the reported latency characteristics from the client devices **350**, the host's media application **322** determines a maximum latency of all client devices **350** in the group being used for playback. This maximum latency is then added to the playback timeline **329**.

XI. Synchronized Local Playback at Host Device

[**0100**] In addition to synchronized playback at multiple client devices **350**, the disclosed system **300** allows for synchronized local playback at the host device **320** running the media application **322**. For example, the host device **320** can play the same audio to its local speakers (not shown) that is being played by the client devices **350**, and the host device **350** can have that same audio play in sync with the all the other devices **350**. To achieve this, the host device **320** uses many of the same principles as applied to the client devices **350**. Rather than receiving packets of audio data over a wireless network, however, audio data is delivered directly to a local playback engine **323** of the media application **322**. In addition, because local playback on the host device **320** is handled by the media application **322**, there is no need for the host device **320** to synchronize time with its own reference clock **324**.

[**0101**] The packets of audio data delivered to the synchronized local playback engine **323** within the media application **322** are generated before being compressed and encrypted. Since these packets do not leave media application **322**, no compression or encryption is necessary. In one embodiment, the host device **320** uses CoreAudio to playback audio. CoreAudio can be used for both Mac-based or Windows-based computers because QuickTime 7 provides support for CoreAudio on Windows-based computers. During operation, an output AudioUnit is opened, and a callback is installed. The callback is called when CoreAudio needs audio data to play. When the callback is called, the media application **322** constructs the relevant audio data from the raw packets delivered to it along with the RTP->NTP timing information. Since CoreAudio has different latency characteristics than the latency characteristics associated with the client devices **350**, information is also gathered about the presentation latency associated with the audio stream of CoreAudio. This information is used to delay the CoreAudio audio stream so that it plays in sync with the known latency of the audio streams associated with the client devices **350**.

XII. Stutter Avoidance During Audio Playback

[**0102**] In addition to the techniques discussed previously for handling lost RTP packets of audio data and for synchronizing clocks between the host device **320** and the client devices **350**, the disclosed system **300** preferably limits stuttering in the playback of media. Referring to FIG. **10**, an algorithm **500** for limiting stutter in the playback of media is shown in flowchart form. This algorithm **500** can be performed by the host device of the disclosed system for each of the client devices. Using the algorithm **500**, the disclosed system detects audible "glitches" caused by gaps in the media (e.g., audio). These gaps can be caused by loss of packets, packets arriving too late, changes to the synchronized timeline, large amounts of clock skew, or other reasons. First, the system determines the number of such "glitches" occurring in a period of time for each of the client devices (Block **502**). Then, a determination is made whether the number of glitches is greater than a predetermined limit (Block **504**). For example, the audio is analyzed over a period of 250-ms to determine whether the 250-ms period is either "glitching" (bad) or "glitch-free" (good). A credit system is used to make this determination. Each time a glitching period is detected, the system takes away a number of credits from a credit score of the client device. The credit score is capped at a minimum value to prevent a long sequence of glitching periods from requiring protracted period of time for the client device to recover, because the intention is to allow the client device to recover quickly as soon as its audio situation clears up.

[0103] If the number of credits goes below a predefined threshold at Block **504**, the client device is put on probation (Block **506**). When on probation, audio is disabled and silenced, but the client device can still send retransmit requests to the host device as needed to recover lost packets of audio data. The audio is silenced during probation so that the client device will not produce an annoying stutter sound when a significant number of glitching periods are successively delivered in an interval of time. Even though the audio is silenced, retransmits remain enabled so that operation of the client device can improve to a point suitable to resume playback.

[0104] If the number of glitches is not greater than the limit at Block **504**, then the client device is set as "glitch free" (Block **505**). Each time a "glitch-free" period is detected, for example, a number of credits is added to the credit score for the client device. The number of credits is capped at a maximum value to prevent a long sequence of glitch-free periods from extending the number of glitches required before going into stutter avoidance mode because the intention is to be able to go into stutter avoidance mode quickly so that there is not any significant stutter produced.

[0105] For the client device on probation with audio silenced and retransmits enabled, the number of glitches occurring in a predetermined unit of time (e.g., X seconds) is determined (Block **508**). The number of glitches is compared to a predetermined limit or threshold (Block **510**). If the client device is on probation for the predetermined unit of time (X seconds) and the number of credits reaches an upper threshold at Block **510**, the client devices is placed back into normal playback mode at Block **505**.

[0106] If the client device remains on probation for the predetermined unit of time (X seconds) and the number of credits has not reached an upper threshold at Block **510**, then the client device is put in jail (Block **512**). When in jail, the audio remains disabled and silenced. However, retransmits are now disabled. In this situation, the client device has not recovered for a significant period of time, and any retransmits may actually be making the situation worse. By disabling retransmits, the recovery time may be improved by reducing congestion on the network. In addition, disabling retransmits may at least reduce the amount of traffic on the network and may allow other client devices to receive packets of audio data more reliably.

[0107] If the client device remains in jail for a predetermined unit of time (e.g., Y seconds) at Block **514**, the client device goes on parole to see if its situation has improved (Block **516**). When on parole, audio is still disabled and silenced. However, retransmits are re-enabled. The number of glitches occurring in a predetermined unit of time (e.g., Z seconds) is determined (Block **518**) and compared to a predetermined limit (Block **520**). If the client device is on parole for the predetermined unit of time and the number of credits reaches an upper threshold at Block **520**, then client device returns to normal playback mode at Block **505** where audio and retransmits are both enabled. If the client device stays on parole for the predetermined unit of time and the number of credits does not reach the upper threshold at Block **520**, however, the client device goes back to jail at Block **512**.

XIII. Handling Address Resolution Protocol

[0108] With reference again to FIG. **5**A, for example, the high volume of data being exchanged by the disclosed system **300** can cause Address Resolution Protocol (ARP) requests, which are broadcast, to become lost. This may be the case especially when the ARP requests are wirelessly broadcast. Address Resolution Protocol (ARP) is a network protocol used to map a network layer protocol address to a data link layer hardware address. For example, ARP can be used to resolve an IP address to a corresponding Ethernet address. When ARP requests are lost, ARP entries at the host device **320** can expire and can fail to be renewed during operation of the disclosed system **300** so that connections between the host device **320** and client devices **350** may appear to go down. Because steady, unicast streams **310** of packets **330** are being exchanged during operation of the disclosed system **300**, one solution to this problem is to extend the expiration times of the ARP entries at the host device **320** as long as packets **330** from the host device **320** are being received by the client devices **350**. By extending the expiration time, the ARP entry for a given client device **350** does not time out (as long as packets **330** are being received by that client device **350**), and the client device **350** does not need to explicitly exchange ARP packets, which may tend to get lost as noted previously, with the host device **320**.

[0109] In another solution, the client devices **350** periodically (e.g., once a minute) send unsolicited, unicast ARP request packets (not shown) to the host device **320**. These unicast ARP request packets contain source addresses (Internet Protocol (IP) address and the hardware address of the client device **350**) and target addresses (IP address and hardware address of the host device **320**). The unicast ARP request packets are more reliable than broadcast packets because the unicast packets are acknowledged and retried at a wireless layer. To keep the ARP entries on the host device **320** for the client devices **350** from expiring, the host device **320** updates its ARP cache when it receives these unicast ARP request packets by refreshing the timeout for the corresponding ARP entries. This prevents the host device **320** from needing to issue a broadcast ARP request when the ARP entry for a client device **350** expires because the ARP entries effectively never expire as long as the client devices **350** unicast ARP request packets to the host device **320**.

[0110] The foregoing description of preferred and other embodiments is not intended to limit or restrict the scope or applicability of the inventive concepts conceived of by the Applicants. In exchange for disclosing the inventive concepts contained herein, the Applicants desire all patent rights afforded by the appended claims. Therefore, it is intended that the appended claims include all modifications and alterations to the full extent that they come within the scope of the following claims or the equivalents thereof.

What is claimed is:

1. A host device, comprising:

a network interface for network communication; and

a reference clock,

wherein the host device is configured to:

establish a first network communication link with a first client device;

send media data to the first client device via the first network communication link, the media data having a presentation timeline based on the reference clock, the presentation timeline specifying when to present the media data;

synchronize a local clock of the first client device with the reference clock; and

control processing of media data at the first client device such that the first client device presents processed media according to the presentation timeline.

2. The host device of claim 1, wherein the host device comprises a personal computer.

3. The host device of claim 1, wherein to send media data to the first client device via the first network communication link, the host device is configured to send a unicast stream of packets containing media data to the first client device, each packet having a timestamp at which to present the media data of the packet.

4. The host device of claim 3, wherein the packets comprise Real-Time Transport Protocol encapsulated in User Datagram Protocol packets.

5. The host device of claim 3, wherein the host device is configured to send the packets containing media data at substantially regular intervals.

6. The host device of claim 5, wherein the substantially regular intervals have a fixed delay based on a processing rate used by an engine of the first client device.

7. The host device of claim 1, wherein to synchronize the local clock of the first client device with the reference clock, the host device is configured to send packets to the first client device in response to requests from the first client device, the packets containing time information for correlating the local clock with the reference clock.

8. The host device of claim 7, wherein the packets comprise Network Time Protocol (NTP) encapsulated in Real-Time Transport Control Protocol (RTCP) packets.

9. The host device of claim 8, wherein the packets contain a Real-Time Transport Protocol (RTP) timestamp at which the request form the host device is transmitted, a first NTP timestamp at which the request from the first client device originated, a second NTP timestamp at which the request from the first client device is received by the host device, and a third NTP timestamp at which the response from the host device is transmitted.

10. The host device of claim 1, wherein to control processing of media data, the host device is configured to adjust a presentation time of processed media at the first client device based on a maximum latency value determined for the first client device.

11. The host device of claim 1, wherein the host device is configured to:

establish a second network communication link with a second client device;

send media data to the second client device via the second network communication link, the media data having a presentation timeline based on the reference clock, the presentation timeline specifying when to present the media data;

synchronize a local clock of the second client device with the reference clock; and

control processing of media data at the second client device such that the second client device presents processed media in a synchronized manner commensurate with the first client device.

12. The host device of claim 1, wherein the host device further comprises:

an engine for processing the media data into processed media; and

a media interface for presenting processed media,

wherein the host device is further configured to control processing of media data at the host device such that the host device presents processed media in a synchronized manner commensurate with the first client device.

13. A host device, comprising:

a network interface for network communication,

wherein the host device is configured to:

establish a network communication link with each of a plurality of client devices;

send media data to each of the client devices via the network communication links; and

control processing of media data at each of the client devices such that processed media is presented in a synchronized manner at each of the client devices.

14. The host device of claim 13, wherein the host device comprises a personal computer.

15. The host device of claim 13, wherein to send media data to each of the client devices via the network communication links, the host device is configured to send a unicast stream of packets containing media data to each of the client devices, each packet having a timestamp at which to present the media data of the packet.

16. The host device of claim 15, wherein the packets comprise Real-Time Transport Protocol encapsulated in User Datagram Protocol packets.

17. The host device of claim 15, wherein the host device is configured to send the packets containing media data at substantially regular intervals.

18. The host device of claim 17, wherein the substantially regular intervals have a fixed delay based on a processing rate used by the engines of the client devices.

19. The host device of claim 13, wherein:

the host device comprises a reference clock,

each of the client devices comprises a local clock, and to control processing of media data, the host device is configured to synchronize each of the local clocks with the reference clock.

20. The host device of claim 19, wherein to synchronize each of the local clocks with the reference clock, the host device is configured to send packets to each of the client devices in response to requests from each of the client devices, the packets containing time information for correlating the local clocks with the reference clock.

21. The host device of claim 20, wherein the packets comprise Network Time Protocol (NTP) encapsulated in Real-Time Transport Control Protocol (RTCP) packets.

22. The host device of claim 21, wherein the packets contain a Real-Time Transport Protocol (RTP) timestamp at which the request form the host device is transmitted, a first NTP timestamp at which the request from the client device originated, a second NTP timestamp at which the request from the client device is received by the host device, and a third NTP timestamp at which the response from the host device is transmitted.

23. The host device of claim 13, wherein to control processing of media data, the host device is configured to adjust a presentation time of processed media at each of the client devices based on a maximum latency value determined for the client devices.

24. The host device of claim 13, wherein the host device further comprises:

an engine for processing the media data into processed media; and

a media interface for presenting processed media,

wherein the host device is further configured to control processing of media data at the host device such that the host device presents processed media in a synchronized manner commensurate with the plurality of client devices.

25. A client device, comprising:

a network interface for network communication;

an engine for processing media data into processed media; and

a media interface for presenting processed media,

wherein the client device is configured to:

receive media data via a network communication link;

receive synchronized timing information via the network communication link; and

process media data with the engine such that the client device presents processed media according to the synchronized timing information received.

26. The client device of claim 25, wherein the network interface of the client device comprises a wireless network interface.

27. The client device of claim 25, wherein the client device comprises an integrated wireless network access point having a power adapter, a wireless network interface, a wired network interface, and a multimedia interface encapsulated within a single integrated casing.

28. The client device of claim 25, wherein the client device comprises an entertainment device connected thereto for presenting processed media, wherein the entertainment device is selected from the group consisting of an amplifier, a powered speaker, a stereo system, a video cassette recorder, a DVD player, a television, home theatre system, a laptop, and a personal computer.

29. The client device of claim 25, wherein to receive media data, the client device is configured to receive a unicast stream of packets containing media data, each packet having a timestamp and a sequence number.

30. The client device of claim 29, wherein the packets comprise Real-Time Transport Protocol encapsulated in User Datagram Protocol packets.

31. The client device of claim 29, wherein the client device is configured to queue packets for processing by the engine based on the sequence numbers.

32. The client device of claim 29, wherein the client device is configured to:

determine whether a packet is missing based on the sequence numbers; and

send a request for retransmission of a packet determined missing.

33. The client device of claim 25, wherein the client device comprises a local clock, and wherein to process media data with the engine according to the synchronized timing information, the client device is configured to synchronize the local clock based on the synchronized timing information.

34. The client device of claim 33, wherein to synchronize the local clock based on the synchronized timing information, the client device is configured to:

send requests for synchronized timing information; and

receive responses to the requests, the responses having packets containing time information for correlating the local clock with a reference clock.

35. The client device of claim 34, wherein the packets comprise Network Time Protocol (NTP) encapsulated in Real-Time Transport Control Protocol (RTCP) packets.

36. The client device of claim 35, wherein the packets contain a Real-Time Transport Protocol (RTP) timestamp at which the request form the host device is transmitted, a first NTP timestamp at which the request from the client device originated, a second NTP timestamp at which the request from the client device is received by the host device, and a third NTP timestamp at which the response from the host device is transmitted.

37. The client device of claim 33, wherein to present the processed media in a synchronized manner, the client device is configured to:

construct media data in received packets into a presentation timeline; and

process media data in the presentation timeline with the engine based on timestamps in the packets correlated to the synchronized time of the local clock of the client device.

38. A network media system, comprising:

means for transferring media data between a plurality of devices;

means for transferring time information between the devices;

means for synchronizing clocks at each of the devices based on the time information;

means for processing transferred media data into processed media at each of the devices; and

means for presenting processed media in a synchronized manner at each of the devices.

39. A network media presentation method, comprising the steps of:

identifying at least one client device connected to a network, the at least one client device capable of processing media data into processed media;

establishing a communication link with the at least one client device via the network;

sending media data to the at least one client device via the communication link;

processing media data into processed media at the at least client device; and

controlling processing of media data at the at least one client device such that processed media is presented in a synchronized manner commensurate with at least one other device.

40. The method of claim 39, wherein the step of identifying comprises receiving user input selecting which identified devices are designated to be a destination for presenting media.

41. The method of claim 39, wherein the step of establishing the communication link comprises establishing the communication link between a host device and the at least one client device.

42. The method of claim 39, wherein the step of sending media data comprises sending a unicast stream of packets containing media data to the at least one client device, each packet having a timestamp and a sequence number.

43. The method of claim 42, wherein the step of processing media data comprises queuing packets for processing by an engine at the at least one client device based on the sequence numbers.

44. The method of claim 42, wherein the step of processing media data comprises:

determining at the at least one client device whether a packet is missing based on the sequence numbers; and

sending a request for retransmission of a packet determined missing.

45. The method of claim 44, wherein the step of sending media data further comprises sending a response to the request, the response containing the packet determined missing.

46. The method of claim 42, wherein the step of sending media data comprises sending the packets containing media data at substantially regular intervals.

47. The method of claim 39, wherein:

each of the devices comprises a clock, and

the step of control processing of media data comprises synchronizing each of the clocks with a reference clock.

48. The method of claim 47, wherein the step of synchronizing comprises sending packets to each of the devices, the packets containing time information for correlating the local clocks with the reference clock.

49. The method of claim 47, wherein the step of processing media data comprises the steps of:

constructing media data contained in received packets into a presentation timeline; and

processing media data in the presentation timeline based on timestamps in the packets correlated to the synchronized time of the local clock of the device.

50. The method of claim 39, wherein the step of controlling processing of media data comprises adjusting a presentation time of processed media at each of the devices based on a maximum latency value determined for the devices.

51. The method of claim 39, wherein the step of controlling processing of media data comprises:

determining a number of gaps in media occurring at each of the devices; and

disabling presentation of processed media or retransmission of lost packets of media data at a given device based on the number of gaps determined for that device.

52. An electronically readable medium having instructions encoded thereon executable by a device for performing a network media presentation method, the method comprising the steps of:

identifying at least one client device connected to a network, the at least one client device capable of processing media data into processed media;

establishing a communication link with the at least one client device via the network;

sending media data to the at least one client device via the communication link;

processing media data into processed media at the at least client device; and

controlling processing of media data at the at least one client device such that processed media is presented in a synchronized manner commensurate with at least one other device.

53. The medium of claim 52, wherein the step of identifying comprises receiving user input selecting which identified devices are designated to be a destination for presenting media.

54. The medium of claim 52, wherein the step of establishing the communication link comprises establishing the communication link between a host device and the at least one client device.

55. The medium of claim 54, wherein the step of sending media data comprises sending a unicast stream of packets containing media data from the host device to the at least one client device, each packet having a timestamp and a sequence number.

56. The medium of claim 55, wherein the step of processing media data comprises queuing packets for processing by an engine at the at least one client device based on the sequence numbers.

57. The medium of claim 55, wherein the step of processing media data comprises:

determining the at least one client device whether a packet is missing based on the sequence numbers; and

sending a request for retransmission of a packet determined missing.

58. The medium of claim 57, wherein the step of sending media data further comprises sending a response to the request, the response containing the packet determined missing.

59. The medium of claim 55, wherein the step of sending media data comprises sending the packets containing media data at substantially regular intervals.

60. The medium of claim 52, wherein:

each of the devices comprises a clock, and

the step of control processing of media data comprises synchronizing each of the clocks with a reference clock.

61. The medium of claim 60, wherein the step of synchronizing comprises sending packets to each of the devices, the packets containing time information for correlating the local clocks with the reference clock.

62. The medium of claim 60, wherein the step of processing media data comprises the steps of:

constructing media data contained in received packets into a presentation timeline; and

processing media data in the presentation timeline based on timestamps in the packets correlated to the synchronized time of the local clock of the device.

63. The medium of claim 52, wherein the step of controlling processing of media data comprises adjusting a presentation time of processed media at each of the devices based on a maximum latency value determined for the devices.

64. The medium of claim 52, wherein the step of controlling processing of media data comprises:

determining a number of gaps in media occurring at each of the devices; and

disabling presentation of processed media or retransmission of lost packets of media data at a given device based on the number of gaps determined for that device.

* * * * *