(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) **World Intellectual Property Organization**
International Bureau

(43) **International Publication Date**
23 October 2014 (23.10.2014)

WIPO | PCT

(10) **International Publication Number**
WO 2014/170715 A1

(51) **International Patent Classification**:
H04L 29/06 (2006.01)

(21) **International Application Number**:
PCT/IB2013/002615

(22) **International Filing Date**:
22 November 2013 (22.11.2013)

(25) **Filing Language**: English

(26) **Publication Language**: English

(30) **Priority Data**:
13305502 17 April 2013 (17.04.2013) EP

(71) **Applicant**: **THOMSON LICENSING** [FR/FR]; 1 rue Jeanne d'Arc, F-921443 Issy-les-Moulineaux (FR).

(72) **Inventor; and**
(71) **Applicant** : **CHAMPEL, Mary-Luc** [FR/FR]; 4, rue de la Croix Beucher, F-35200 Marpier (FR).

(81) **Designated States** *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*

(54) **Title**: METHOD AND APPARATUS FOR PACKET HEADER COMPRESSION

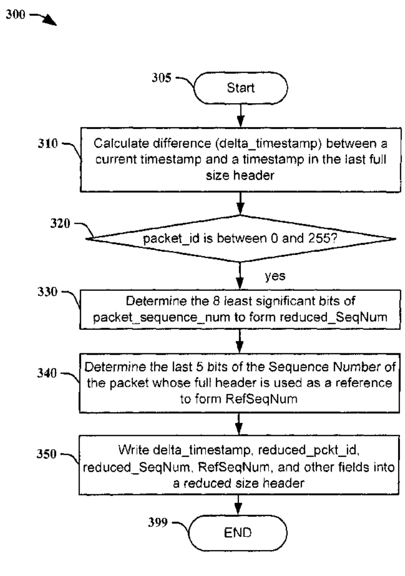

FIG. 3

(57) **Abstract**: To compress headers efficiently, headers may be differentially encoded. In one embodiment, a difference between a current field and a reference field can be used in a reduced header. In another embodiment, the least significant bits of the current field can be used to represent the current field. When such a field is received, the most significant bits of a reference field are needed to de¬ compress the field. In addition, the present principles determine the typical usage scenarios that may benefit from header compression and further determine that some fields can be removed in a reduced header or be represented by fewer bits. Header compression may impose constraints on the representation of the current field. The present principles recognize constraints on using reduced headers and provide rules and guidelines on applying header compression.

WO 2014/170715 A1

Method and Apparatus for Packet Header Compression

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims the benefit of the filing date of EP Application number EP13305502.0, which is hereby incorporated by reference in its entirety for all purposes, filed on April 17, 2013, and titled “Method and Apparatus for Packet Header Compression.”

TECHNICAL FIELD

This invention relates to a method and an apparatus for generating a compressed transport packet header, and a method and apparatus for processing the same.

BACKGROUND

In communication networks, one of the most expensive resources is bandwidth. Audio and video compression methods have been developed to effectively reduce the amount of data to be transmitted over the communication networks. In addition, it is important to reduce the overhead introduced by the use of transport protocols. More specifically, it is important to reduce the size of transport packet headers.

SUMMARY

The present principles provide a method of transmitting data, comprising the steps of: encapsulating at least one of video data and audio data in a datastream comprising a plurality of transport packets, each transport packet comprising a header portion and a payload portion; determining whether a particular transport packet can be transmitted using one of a full size packet header format and a reduced sized packet header format; forming transport packets for the datastream in response to the determining step; and transmitting the formed transport packets as described below. The present principles also provide an apparatus for performing these steps.

The present principles also provide a method of receiving data, comprising the steps of: receiving a datastream, which includes at least one of video data and audio data, comprising a plurality of transport packets, each transport packet comprising a header portion and a payload portion; determining whether a particular transport packet uses one of a full size packet header format and a reduced sized packet header format; and decoding transport packets for the datastream in response to the determined format as described below. The present principles also provide an apparatus for performing these steps.

The present principles also provide a computer readable storage medium having stored thereon instructions for transmitting or receiving data, according to the methods described above.

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1A is a pictorial example depicting structure of MMT (MPEG Multimedia Transport) payload header followed by payload data, and FIG. 1B is a pictorial example depicting structure of MMT payload header extension.

FIG. 2 is a pictorial example depicting structure of MMT packet header.

FIG. 3 is a flow diagram depicting an exemplary method for generating a reduced header, in accordance with an embodiment of the present principles.

FIG. 4 is a pictorial example depicting that bits in a sequence number of a full sized header are used to generate `reduced_SeqNum` for a reduced header, and that `reduced_SeqNum` is used to decode a sequence number for the reduced header, in accordance with an embodiment of the present principles.

FIG. 5A is a pictorial example depicting structure of a modified MMT packet with a full header, and FIG. 5B is a pictorial example depicting structure of a modified MMT packet with a reduced header, in accordance with an embodiment of the present principles.

FIG. 6A is a pictorial example depicting structure of a modified MMT payload header followed by payload data, and FIG. 6B is a pictorial example depicting structure of a reduced MMT payload header followed by payload data, in accordance with an embodiment of the present principles.

FIG. 7 is a flow diagram depicting an exemplary method for determining whether to use a full header or a reduced header, in accordance with an embodiment of the present principles.

FIG. 8 is a flow diagram depicting an exemplary method for de-compressing a reduced header, in accordance with an embodiment of the present principles.

FIG. 9 is a block diagram depicting an exemplary transmitting system, in accordance with an embodiment of the present principles.

FIG. 10 is a block diagram depicting an exemplary receiving system, in accordance with an embodiment of the present principles.

FIG. 11 is a block diagram depicting another exemplary receiving system, in accordance with an embodiment of the present principles.

DETAILED DESCRIPTION

MPEG-H part 1 standard (also known as MPEG Multimedia Transport or MMT) defines a complete solution for packaging, transport and composition of timed and non-timed media content. MMT is currently under development, with a draft standard described in "Text of ISO/IEC 2nd CD 23008-1 MPEG Media Transport, MPEG/N13293," Geneva, Switzerland, January 2013 (hereinafter "MMT_CD").

While MMT primarily addresses IP networks, it also supports delivery of content over any type of packet-based networks. In particular, MMT may be used for delivery of audiovisual services over broadcast networks such as terrestrial, cable or satellite networks.

In MMT, the term "Asset" refers to a data entity containing data with the same transport characteristics and that is composed of one or more MPUs (media processing units) with same Asset ID, the term "Package" refers to a logical collection of data, which is composed of one or more Assets and their related Asset Delivery Characteristics (i.e., description about required Quality of Service for delivery of Assets), and an Composition Information (i.e., description of spatial and temporal relationship among Assets).

MMT packaging and transport capabilities are defined in two protocols, namely, MMT-Payload Format (MMT-PF) and MMT-Transport Protocol (MMT-TP). Specifically, MMT-PF defines a generic payload format for packetizing content components (for example, audio, video, and files) of a multimedia package. MMT-PF is agnostic to specific media codecs used for coding media data, and is also used to packetize signaling messages and Forward Error Correction (FEC) information. MMT payload format can be used for any packet-based transport protocols such as RTP or MMT transport protocol. MMT-TP defines a transport protocol supporting streaming delivery of Package through packet-based heterogeneous delivery network including IP network environments. The MMT protocol provides essential features for delivery of Package such as protocol level multiplexing that enables various Assets to be delivered over a single MMT packet flow, delivery timing model independent of presentation time to adapt to a wide range of network jitter, and information to support Quality of Service.

FIGs. 1A and 1B show the structure of MMT payload header followed by payload data and structure of MMT payload header extension, respectively, according to the MMT-PF protocol. In the present application, the payload header is also referred to as MMT-PF header. A complete description of semantics can be found in MMT_CD. The semantics of some fields shown in FIGs. 1A and 1B, which may be removed or modified according to the present principles, are reproduced below.

`length (16bits)` – This field indicates the length of the payload in bytes excluding the header. This doesn't include the size of padding data.

`f_i (2bits)` – This field indicates the fragmentation indicator contains information about fragmentation of data unit in the payload.

`fragmentation_flag (F: 1bit)`- Set to '1', if `fragment_counter` is present.

`aggregation_flag (A: 1bit)` - Set to '1', if `aggregation_info` is present.

`RAP_flag (R: 1bit)` - Set to '1', if payload contains random access point (or part thereof).

`payload_sequence_flag (P: 1bit)` - Set to '1', if `payload_sequence_number` is present.

`number_data_unit (numDU: 4 bits)` - This field specifies the number of data unit within this MMT payload. This field shall be '0', if `fragmentation_flag` is set to '1'.

`DU_offset (16bits)` – This field specifies location of each data unit from the byte indicated by `data_offset`. This field shall be used, when `aggregation_flag` is set to '1'.

`payload_sequence_number (32 bits)` – This field specifies the sequence number of payload associated with the same Asset.

FIG. 2 shows the structure of MMT packet header according to the MMT-TP protocol. A complete description of semantics can be found in MMT_CD. In the present application, the MMT packet header is also referred to as MMT-TP header. The semantics of some fields shown in FIG. 2, which may be removed or modified according to the present principles, are reproduced below.

`packet_id (16bits)` - This field is an integer value assigned to each Asset to distinguish packets of one Asset from another.

`packet_sequence_number (32bits)` - This field is an integer value that is scoped by the `packet_id` and that starts from arbitrary value incremented by one for each MMT packet. It wraps around to '0' after its maximum value.

`timestamp (32bits)` - This field specifies the time instance of MMT packet delivery. The NTP (Network Time Protocol) time is used in timestamp as specified as the "short-format" in clause 6 of IETF RFC5905, NTP version 4. This timestamp is measured at the first bit of MMT packet.

Both MMT-PF and MMT-TP include variable size headers with a minimum size of 9 bytes for MMT-PF (with usual presence of sequence number) and 12 bytes (and 3 bits) for MMT-TP. Since MMT-PF and MMT-TP may be used to transport very small packet payloads, those extra 21 bytes spent on headers may represent a very high overhead.

In order to reduce the size of headers, technique such as Robust Header Compression (RoHC defined in RFC 3095) may be used. While RoHC can effectively reduce the size of headers, it relies on complex coding techniques that may require heavy computation on the receiver side and it forces the receiver to decode all headers even when the receiver only needs to examine a portion of the headers for packet filtering purposes.

The present principles provide a method and apparatus that address the issues discussed above and can significantly reduce the header size with a low computational complexity.

In one embodiment, a header may be sent with or without compression. In the present application, a header without compression is referred to as a "full size header" or "full header," and a header that is compressed is referred to as a "reduced size header" or "reduced header." To differentiate whether header compression is used, a field, designated as a `C` field in the present embodiments, can be added, for example, at the beginning of the header. In one example, when `C` is set to 0, the header includes a "full size" header, and when `C` is set to 1, the header includes a "reduced size" header.

To form a reduced size header, some fields in a full size header may be removed, some fields may be represented with fewer bits than the full header, and new fields may be added. The order of the fields in the header may also be changed. In the following, exemplary embodiments of header compression for MMT-TP and MMP-PF are discussed in further detail.

MMT-TP header

FIG. 3 illustrates an exemplary method 300 for generating a reduced MMT-TP header. Method 300 starts at step 305. At step 310, a `delta_timestamp` field is used to represent the timestamp. Although it is costly in terms of bits, the timestamp information associated with a packet needs to be preserved.

The `delta_timestamp` field contains the difference between the `timestamp` field of a reference full size header and the value that would be in the current `timestamp` field if a full header were used. In one embodiment, a full size header may be inferred to be a reference header based on rules known to both the receiver and transmitter, for example, the last received full size header is used as a reference header. In another embodiment, a full size header may be indicated explicitly to be a reference header. The difference between timestamps is coded in a way similar to the 19 least significant bits of an NTP timestamp. This keeps the same timestamp precision with a duration of 8 seconds. If the difference between these two timestamps is larger than 8 seconds (and therefore goes beyond the maximum duration that can be coded with 19 bits), then a packet with a full header is sent in order to provide a new timestamp reference value for further packets with reduced size headers.

The `packet_id` field in a full header is replaced by a `reduced_pckt_id` field. The `reduced_pckt_id` field uses 8 bits rather than 16 bits in the full header. Therefore, this reduction from 16-bit packet ids to 8-bit ids restricts the use of the reduced header for streams whose `packet_id` is between 0 and 255. Thus, at step 320, it checks whether `packet_id` is between 0 and 255. If yes, method 300 sets a value for `reduced_pckt_id` and continues with the generation of the reduced header. If not, the proposed header reduction mechanism cannot be used and a full header should be generated.

At step 330, a `reduced_SeqNum` field is used to represent the packet sequence number. The `reduced_SeqNum` field contains the 8 least significant bits of the `packet_sequence_number` field that would be in the header if a full size header were used. Since this new field is coded by 8 bits, the present principles mandate that for each stream (identified by the same packet_id) a packet with a full size header shall be sent at least every 256 packets.

At step 340, a new field, RefSeqNum is generated. RefSeqNum contains the last 5 bits of the Sequence Number of the packet whose full header is used as a reference. This new field brings additional robustness by allowing the receiver to check if the last full header received is actually the one that shall be used as a reference for the current reduced size header. Since a packet may be dropped, field RefSeqNum provides a mechanism to make sure that when a full header reference has not been received, the receiver does not try to improperly decode the reduced header.

At step 350, fields "delta_timestamp," "reduced_pckt_id," "reduced_SeqNum," "RefSeqNum," and other field are written into the reduced size header, for example, according to a format shown in FIG. 5B. Method 300 ends at step 399.

Note that the steps in method 300 may proceed at a different order from what is shown in FIG. 3, for example, steps 310-340 may be performed in any order.

FIG. 4 illustrates that bits in a sequence number are used to generate field "reduced_SeqNum," and field "reduced_SeqNum" is used to decode the sequence number in a reduced header using an example. In this example, current packet N+i is to be sent, and a last full size header is received in packet N. At the transmitter side, only the 8 least significant bits (430) of the sequence number (415) are kept to generate field "reduced_SeqNum", which reduces the required processing.

Decoding at the receiver side is more complex than encoding at the transmitter side. First, the de-compressor needs to store the sequence number (405), or the 24 most significant bits of the sequence number (410), of the packet that is used as a reference. The de-compressor may also need to store the 8 least significant bits (430) of the sequence number as an initial value for reduced_SeqNum. Then for each received reduced header the decoder needs to track if the reduced_SeqNum is looped through 0. The complete sequence number of a reduced header packet is then obtained by appending the reduced_SeqNum (430) to the 24 most significant bits (440) of the reference sequence number (incremented by 1 if reduced_SeqNum has been looped through 0). That is, the bits shown in 440 equal the bits shown in 410 when reduced_SeqNum has not been looped through 0, and is greater than the bits shown in 410 by 1 otherwise.

FIG. 5A illustrates the proposed full MMT-TP header, and FIG. 5B illustrates the proposed reduced MMT-TP header, according to the present principles. Compared with the MMT-TP header shown in FIG. 2, fields "Q," "F," "P," "FEC," "RES," "TB," "DS," and "R" are moved to the beginning of the proposed full MMT-TP header and their sizes and semantics are not changed, field "S" is removed, and fields "C" and "I" are added. "I" flag is used to indicate whether the current header information shall be stored since it will be later used as a reference, and it has been added to full MMT-PF and MMT-TP headers.

The order of the fields is also adjusted. Since flag "C" indicates what kind of header will be used, it needs to be the first information determined by the decoder. Assuming a decoder will determine first bit first, using the first bit for flag "C" enables the decoder to first determine what kind of header will follow.

Comparing the proposed reduced MMT-TP header as shown in FIG. 5B with the full MMT-TP header as shown in FIG. 5A, fields "packet_id," "packet_sequence_number," and "timestamp" have been replaced with "reduced_pckt_id," "reduced_SeqNum," and "delta_timestamp," respectively, field "RefSeqNum" has been added, and fields "I," "RES," and "reserved" have been removed. Consequently, the minimum MMT-TP header size is decreased from 99 bits to 56 bits, which represents a 43% bit saving.

MMT-TP streams are identified by their packet_id and each stream has its own packet_sequence_number sequence. Therefore, by applying the header reduction mechanism on a limited number of streams (packet_id between 0 and 255), the combination of RefSeqNum and reduced_pckt_id fields identifies in a unique way the reference packet that shall be used for the decoding of a reduced size header.

Moreover, since reduced_pckt_id is nothing but a copy of packet_id field with a smaller range of possible values, traditional packet filtering usually achieved by filtering on packet_id can be operated the same way on reduced_pckt_id. In one example, filtering consists in looking at some fields in order to decide whether or not a packet should be transmitted or received. When using techniques such as RoHC, in order to read one field of the header, the whole header needs to be uncompressed. By contrast, with our proposed technique, only the desired field needs to be uncompressed. That is, filtering can be done directly on reduced_packet_id without reconstructing original packet_id. Therefore the header compression technique according to the present principles is totally transparent on packet filtering.

MMT-PF header

Header compression usually offers interest for large and repetitive headers. Specific to MMT, header compression may provide significant bit savings for FEC repair symbols or when a single MPU is transported in several fragments. Thus, we design the reduced header for some specific usage scenarios that may benefit from header compression. Considering the usage scenarios, we propose a reduced size MMT-PF header in the following.

(1) The A, number_data_unit, and DU_offset fields are removed since aggregation is not supported by a reduced size header. Consequently full headers shall be used when aggregation is used.

(2) The R flag is removed since a full header shall always be used when a Random Access Point (RAP) is present in the payload. This ensures that packets with RAP can be decoded on their own (without relying on the encoding of other "reference" packet).

(3) The P and payload_sequence_number fields have been removed as all packets with reduced headers shall share the same payload_sequence_number of their "reference" packet.

(4) The length field is removed as header compression is used only on same size fragments. Consequently, the first fragment shall use a full header and any fragment with a size different from "reference" fragment (usually first fragment, but it is not mandatory) shall use a full header too.

(5) The RefSNum field contains the last 4 bits of the Payload Sequence Number of the packet whose full header is used as a reference. This brings additional robustness by allowing the receiver to check if the last full header received is actually the one that shall be used as a reference for the current reduced size header. Since a packet may be dropped, this provides a mechanism to make sure that when a full header reference has not been received, the receiver does not try to improperly decode the reduced header.

FIG. 6A illustrates the proposed full MMT-PF header, and FIG. 6B illustrates the proposed reduced MMT-PF header. Compared with the MMT-PF header shown in FIG. 1A, fields "f_i," "A," "R," "P," and "E" are moved to the beginning of the proposed full MMT-PF header, and their sizes and semantics do not change. Flags "F" and "S" are removed, and fields "C" and "I" are added. The order of the fields is also adjusted. Comparing the proposed reduced MMT-PF header as shown in FIG. 6B with the full MMT-PF header as shown in FIG. 6A, fields "I," "A," "R," "P," "length," "numDU," "DU_offset," and "payload_sequence_number" are removed, and field "RefSNum" is added. Consequently, the minimum MMT-PF header size is decreased from 128 bits to 32 bits, which represents a 75% bit saving.

In many applications, all fragments except for the last fragment of a same asset will have the same size and therefore a full size header can be used only in first and last fragments and all other fragments may use reduced headers while using the first fragment as a reference.

As discussed above, different methods have been used to provide efficient and robust header compression for MMT-PF and MMT-TP. In one embodiment, a difference between the current field and the reference field, for example, delta_timestamp, is used in a reduced header. By using the difference rather than the field itself, fewer bits can be used to represent the field. In another embodiment, the least significant bits of the current field, for example, reduced_SeqNum, is used to represent the current field. When such a field is received, the most significant bits of a reference field is needed to de-compress the field. These two embodiments, that is, using a difference and using the least significant bits, both represent the current field differentially to achieve header compression efficiency. In another embodiment, an arrangement according to the present principles determines the typical usage scenarios that may benefit from header compression and further determines that some fields can be removed in a reduced header. In addition, fewer bits are used to represent a current field (for example, reduced_pckt_id) in a reduced header, which may impose constraints on the values that can be represented by the current field. The present principles recognize constraints on using reduced headers and provide rules and guidelines on setting values for the fields.

As discussed above, there are rules and constraints for using reduced headers. FIG. 7 illustrates an exemplary method 700 for determining whether to use a reduced header or a full size header considering the rules and constraints. This method is performed, for example, in an encoder that encodes the headers associated with a transport stream according to MMT-TP. Method 700 starts at step 710 that performs initialization.

At step 720, it checks whether the difference between the timestamp of the current packet and the timestamp of the reference packet is larger than 8 seconds (and therefore cannot be coded on the 19-bit `delta_timestamp` field). If yes, method 700 generates a full size header at step 770.

At step 730, it checks whether `packet_id` is in the range between 0 and 255. If not, the value of `packet_id` is beyond what can be properly represented by `reduced_pckt_id` and method 700 generates a full size header at step 770.

At step 740, it checks for each stream (identified by `packet_id`) whether the `reduced_SeqNum` is wrapping up to its initial value. If yes, method 700 generates a full size header at step 770 to provide a reference sequence number for further packets using reduced headers.

At step 750, it checks whether the packet is the first packet of an access unit which contains a Random Access Point (RAP). If yes, method 700 generates a full size header at step 770. Otherwise, a reduced header is generated at step 760.

The steps in method 700 may proceed at a different order from what is shown in FIG. 7, for example, steps 720-750 may be performed in any order. Method 700 may also choose to generate a full size header in other conditions, for example, based on a user request.

To determine whether a full size header or a reduced header can be used for MMT-PF, the following conditions can be considered:

(1) Aggregation mechanism is used;

(2) A Random Access point (RAP) is present;

(3) A new payload sequence number is used;

(4) The length value of the current packet is different from the length value of the reference packet.

When at least one of the above conditions is met, a full MMT-PF header should be sent. A full MMT-PF header can also be sent in other conditions, for example, based on a user request.

At the receiver side, a receiver performing a method according to the present principles determines whether a full size header or a reduced header is used, for example, using "C" flag. The receiver stores important information, for example, what is required for de-compressing the header, from full headers when such headers are marked as references.

FIG. 8 illustrates an exemplary method 800 for de-compressing a reduced size MMT-PT header. Method 800 starts at step 805. At step 810, it receives a reduced header and parses fields, for example, but not limited to, "delta_timestamp," "reduced_SeqNum," and "RefSeqNum" from the reduced size header. At step 820, it checks whether the last 5 bits of the packet_sequence_number of the full size header used as a reference is the same as RefSeqNum. If they are not the same, the reduced size header cannot be properly decoded and method 800 ends at step 899. Otherwise, at step 830, it determines timestamp and packet_sequence_number of the full size header that is used as a reference for the current reduced size header. At step 840, it determines timestamp for the current packet as a sum of timestamp in the reference full header and delta_timestamp. At step 850, it determines packet sequence number for the current packet based on packet_sequence_number of the full size header and reduced_SeqNum, for example, as illustrated in FIG. 4. Method 800 ends at step 899.

Using MMT-PF and MMT-TP headers as examples, we discuss how various embodiments can be used to efficiently compress the headers. Because of header compression, it is important that a correct reference field is used for proper decoding of a current field. To provide decoding robustness, a link to the reference field (for example, RefSeqNum) can be used to prevent improper decoding of reduced size headers. The present principles can also be used for header compression in other applications or protocols.

In various examples discussed above, specific values, for example, the number of bits for fields and flags, and specific orders of fields are described. These values or orders may need to be adjusted accordingly when the present principles are applied in different applications or transport protocols.

Coded packets in reduced headers according to the present principles can be easily filtered without complex header decoding of all packets. Thus, the present principles provide the advantage of network transparency. In addition, with the use of simple header compression mechanisms, the present principles can be implemented at low computational complexity and do not require significant processing at the receiver.

FIG. 9 illustrates an exemplary transmitting system 900. The input data, for example, but not limited to, audio and video data, are encoded at media encoder 910. The encoded data is multiplexed at multiplexer 920, and transmitted at transmitter 930. The header compression mechanism according to the present principles, for example, methods 300 and 700, can be used in a header compressor (940, 950) that is located in multiplexer 920 or transmitter 930. The transmitting system may be used in a typical broadcast TV environment where bandwidth is an expensive resource, or may be used in a mobile device that provides audiovisual service. By using header compression at the multiplexer 920, following the media encoding process, it is possible to prepare header compression in advance since MMT-PF flow may be stored in a file format before actual transmission. At transmitter 930, the system may re-use MMT-PF header compression (with possible updates) and also use MMT-TP header compression before sending the actual MMT-TP packets.

FIG. 10 illustrates an exemplary receiving system 1000. The input data of system 1000 may be a transport bitstream, for example, the output of system 900. The data is received at receiver 1010, de-multiplexed at de-multiplexer 1020, and then decoded at media decoder 1030. When receiving MMT-TP packets, header de-compressor (1040, 1050) performs MMT-TP header de-compression by storing header information from reference packets and decoding reduced headers. Decoded packets can be placed in a buffer of de-multiplexer 1020. When processing MMT-PF payloads, the de-multiplexer 1020 may apply MMT-PF header de-compression.

FIG. 11 illustrates another exemplary receiving system 1100. In overview, in the video receiver system of FIG. 11, a broadcast carrier modulated with signals carrying audio, video and associated data representing broadcast program content is received by antenna 10 and processed by unit 13. The resultant digital output signal is demodulated by demodulator 15. The demodulated output from unit 15 is trellis decoded, mapped into byte length data segments, deinterleaved and Reed-Solomon error corrected by decoder 17. The output data from unit 17 is in the form of an MPEG compatible transport datastream, for example, an MMT transport stream, containing program representative multiplexed audio, video and data components. The transport stream from unit 17 is demultiplexed into audio, video and data components by unit 22 which are further processed by the other elements of decoder 100.

If header compression technique, for example, MMT-PF and MMT-TP header compression, is used, unit 22 performs header decompression before de-multiplexing the stream and sending elementary streams to unit 25, 35, or 95. In one mode, decoder 100 provides MPEG decoded data for display and audio reproduction on units 50 and 55, respectively. In another mode, the transport stream from unit 17 is processed by decoder 100 to provide an MPEG compatible datastream for storage on storage medium 105 via storage device 90.

A user selects for viewing either a TV channel or an on-screen menu, such as a program guide, by using a remote control unit 70. Processor 60 uses the selection information provided from remote control unit 70 via interface 65 to appropriately configure the elements of FIG. 11 to receive a desired program channel for viewing. Processor 60 comprises processor 62 and controller 64. Unit 62 processes (i.e. parses, collates and assembles) program specific information including program guide and system information and controller 64 performs the remaining control functions required in operating decoder 100. Although the functions of unit 60 may be implemented as separate elements 62 and 64 as depicted in FIG. 11, they may alternatively be implemented within a single processor. For example, the functions of units 62 and 64 may be incorporated within the programmed instructions of a microprocessor. Processor 60 configures processor 13, demodulator 15, decoder 17 and decoder system 100 to demodulate and decode the input signal format and coding type.

Considering FIG. 11 in detail, a carrier modulated with signals carrying program representative audio, video and associated data received by antenna 10, is converted to digital form and processed by input processor 13. Processor 13 includes radio frequency (RF) tuner and intermediate frequency (IF) mixer and amplification stages for downconverting the input signal to a lower frequency band suitable for further processing.

It is assumed for exemplary purposes that a video receiver user selects a sub-channel (SC) for viewing using remote control unit 70. Processor 60 uses the selection information provided from remote control unit 70 via interface 65 to appropriately configure the elements of decoder 100 to receive the physical channel corresponding to the selected sub-channel SC.

The output data provided to processor 22 is in the form of a transport datastream containing program channel content and program specific information for many programs distributed through several sub-channels.

Processor 22 matches the Packet Identifiers (PIDs) of incoming packets provided by decoder 17 with PID values of the video, audio and sub-picture streams being transmitted on sub-channel SC. These PID values are preloaded in control registers within unit 22 by processor 60. Processor 22 captures packets constituting the program transmitted on sub-channel SC and forms them into MPEG compatible video, audio streams for output to video decoder 25, audio decoder 35 respectively. The video and audio streams contain compressed video and audio data representing the selected sub-channel SC program content.

Processor 22 also detects whether header compression, for example, MMT-PF and MMT-TP header compression according to the present principles, is used and detects whether a packet provides a reference header for header compression. Processor 22 stores the reference header and uses it to decode reduced size headers.

Decoder 25 decodes and decompresses the MPEG compatible packetized video data from unit 22 and provides decompressed program representative pixel data to device 50 for display. Similarly, audio processor 35 decodes the packetized audio data from unit 22 and provides decoded audio data, synchronized with the associated decompressed video data, to device 55 for audio reproduction.

In a storage mode of the system of FIG. 11, the output data from unit 17 is processed by decoder 100 to provide an MPEG compatible datastream for storage. In this mode, a program is selected for storage by a user via remote unit 70 and interface 65.

Processor 60, in conjunction with processor 22 forms a composite MPEG compatible datastream containing packetized content data of the selected program and associated program specific information. The composite datastream is output to storage interface 95. Storage interface 95 buffers the composite datastream to reduce gaps and bit rate variation in the data. The resultant buffered data is processed by storage device 90 to be suitable for storage on medium 105. Storage device 90 encodes the buffered datastream from interface 95 using known error encoding techniques such as channel coding, interleaving and Reed Solomon encoding to produce an encoded datastream suitable for storage. Unit 90 stores the resultant encoded datastream incorporating the condensed program specific information on medium 105.

The implementations described herein may be implemented in, for example, a method or a process, an apparatus, a software program, a data stream, or a signal. Even if only discussed in the context of a single form of implementation (for example, discussed only as a method), the implementation of features discussed may also be implemented in other forms (for example, an apparatus or program). An apparatus may be implemented in, for example, appropriate hardware, software, and firmware. The methods may be implemented in, for example, an apparatus such as, for example, a processor, which refers to processing devices in general, including, for example, a computer, a microprocessor, an integrated circuit, or a programmable logic device. Processors also include communication devices, such as, for example, computers, cell phones, portable/personal digital assistants ("PDAs"), and other devices that facilitate communication of information between end-users.

Reference to "one embodiment" or "an embodiment" or "one implementation" or "an implementation" of the present principles, as well as other variations thereof, mean that a particular feature, structure, characteristic, and so forth described in connection with the embodiment is included in at least one embodiment of the present principles. Thus, the appearances of the phrase "in one embodiment" or "in an embodiment" or "in one implementation" or "in an implementation", as well any other variations, appearing in various places throughout the specification are not necessarily all referring to the same embodiment.

Additionally, this application or its claims may refer to "determining" various pieces of information. Determining the information may include one or more of, for example, estimating the information, calculating the information, predicting the information, or retrieving the information from memory.

Further, this application or its claims may refer to "accessing" various pieces of information. Accessing the information may include one or more of, for example, receiving the information, retrieving the information (for example, from memory), storing the information, processing the information, transmitting the information, moving the information, copying the information, erasing the information, calculating the information, determining the information, predicting the information, or estimating the information.

Additionally, this application or its claims may refer to "receiving" various pieces of information. Receiving is, as with "accessing", intended to be a broad term. Receiving the information may include one or more of, for example, accessing the information, or retrieving the information (for example, from memory). Further, "receiving" is typically involved, in one way or another, during operations such as, for example, storing the information, processing the information, transmitting the information, moving the information, copying the information, erasing the information, calculating the information, determining the information, predicting the information, or estimating the information.

As will be evident to one of skill in the art, implementations may produce a variety of signals formatted to carry information that may be, for example, stored or transmitted. The information may include, for example, instructions for performing a method, or data produced by one of the described implementations. For example, a signal may be formatted to carry the bitstream of a described embodiment. Such a signal may be formatted, for example, as an electromagnetic wave (for example, using a radio frequency portion of spectrum) or as a baseband signal. The formatting may include, for example, encoding a data stream and modulating a carrier with the encoded data stream. The information that the signal carries may be, for example, analog or digital information. The signal may be transmitted over a variety of different wired or wireless links, as is known. The signal may be stored on a processor-readable medium.

CLAIMS:

1. A method of transmitting data, comprising the steps of:

encapsulating at least one of video data and audio data in a datastream comprising a plurality of transport packets, each transport packet comprising a header portion and a payload portion;

determining whether a particular transport packet can be transmitted using one of a full size packet header format and a reduced sized packet header format;

forming transport packets for the datastream in response to the determining step; and

transmitting the formed transport packets.

2. The method of claim 1, wherein the forming step comprises:

forming a first packet having a first header, wherein the first header is provided as a reference for a second header for a second packet, the first header corresponding to the full size packet header format and the second header corresponding to the reduced sized packet header format; and

generating (310, 330) the second header, wherein a field in the second header is differentially represented with respect to a corresponding field in the first header.

3. A method of receiving data, comprising the steps of:

receiving a datastream, which includes at least one of video data and audio data, comprising a plurality of transport packets, each transport packet comprising a header portion and a payload portion;

determining whether a particular transport packet uses one of a full size packet header format and a reduced sized packet header format; and

decoding transport packets for the datastream in response to the determined format.

4. The method of claim 3, wherein the receiving step comprises: receiving a first packet having a first header; and

receiving (840, 850) a second packet including a second header, the first header corresponding to the full size packet header format and the second header corresponding to the reduced sized packet header format, the second header including a field, and wherein the decoding step comprises:

determining that the first header to be used as a reference for the second header; and

decoding the field in the second header responsive a corresponding field in the first header.

5. An apparatus (900) for transmitting data, comprising:

a transmitter (930) configured to encapsulate at least one of video data and audio data in a datastream comprising a plurality of transport packets, each transport packet comprising a header portion and a payload portion; and

a header compressor (940, 950) configured to determine whether a particular transport packet can be transmitted using one of a full size packet header format and a reduced sized packet header format and to form transport packets for the datastream in response to the determined step, wherein the transmitter is configured to transmit the formed transport packets.

6. The apparatus of claim 5, wherein the transmitter (930) is configured to transmit a first packet having a first header, wherein the first header is provided as a reference for a second header for a second packet, the first header corresponding to the full size packet header format and the second header corresponding to the reduced sized packet header format, and

wherein the header compressor (940, 950) is configured to generate the second header, wherein a field in the second header is differentially represented with respect to a corresponding field in the first header, and wherein the transmitter transmits the second packet including the second header.

7. The method or apparatus of claim 2 or 5, wherein the field in the second header can be represented by a plurality of bits, and wherein the header compressor generates the second header by representing the field in the second header with least significant bits of the plurality of bits.

8. The method or apparatus of claim 2 or 5, wherein the first header is formed when data to be represented in the field exceeds a maximum representation capacity of the number of bits included in the field.

9. An apparatus (1000, 1100) for receiving data, comprising:

a receiver (1010, 10) configured to receive a datastream, which includes at least one of video data and audio data, comprising a plurality of transport packets, each transport packet comprising a header portion and a payload portion; and

a header de-compressor (1040, 1050, 22) configured to determine whether a particular transport packet uses one of a full size packet header format and a reduced sized packet header format and decode transport packets for the datastream in response to the determined format.

10. The apparatus of claim 9, wherein the receiver (1010, 10) is configured to receive a first packet having a first header and receiving a second packet including a second header, the first header corresponding to the full size packet header format and the second header corresponding to the reduced sized packet header format, the second header including a field, and wherein the header de-compressor (1040, 1050, 22) is configured to determine that the first header to be used as a reference for the second header and decoding the field in the second header responsive a corresponding field in the first header.

11. The method or apparatus of claim 2, 4, 6 or 10, wherein the first header includes a second field indicating that the first header is used as the reference.

12. The method or apparatus of claim 4 or 10, wherein the corresponding field in the first header can be represented by a plurality of bits, and wherein the field in the second header is decoded responsive to most significant bits of the plurality of bits and the received field, the decoded field in the second header having the same number of bits used to represent the corresponding field in the first header.

13. The method or apparatus of claim 2, 4, 6 or 10, wherein the second header includes a link to the first packet.

14. The method or apparatus of claim 13, wherein a sequence number in the first header can be represented by a plurality of bits, and wherein the second header includes a portion of the plurality of bits to provide the link to the first packet.

15. The method or apparatus of claim 2, 4, 6 or 10, wherein the second header in the second packet includes fewer fields than the first header in the first packet.

16. A computer readable storage medium having stored thereon instructions for transmitting or receiving data, according to claims 1-4, 7, 8 and 11-15.

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            length             |      type     |f_i|F|A|R|P|E|S|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| data_offset |  frag_count   | numDU |       DU_offset         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ...   |           DU_offset         |           ....          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    payload_sequence_number                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        header_extension                       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          payload data                     ....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 1A

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|             type              |             length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     header_extension_value                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 1B

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           packet_id           |    packet_sequence_number     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    packet_sequence_number     |           timestamp           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           timestamp           |Q|F|P|FEC| RES | TB  | DS  |R|S|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  TP |  flow_label |e| reserved|       private_user_data       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          payload data                     ....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 2

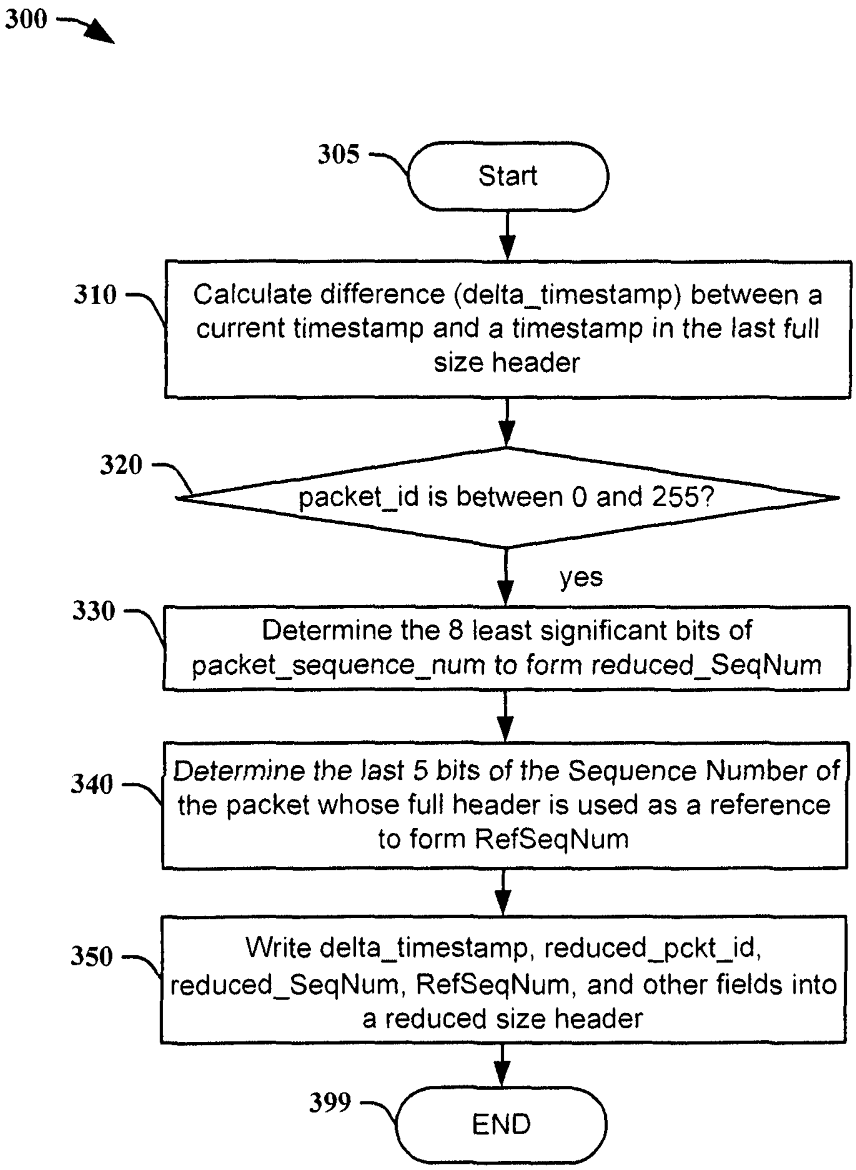
300
305
Start
310
Calculate difference (delta_timestamp) between a current timestamp and a timestamp in the last full size header
320
packet_id is between 0 and 255?
yes
330
Determine the 8 least significant bits of packet_sequence_num to form reduced_SeqNum
340
Determine the last 5 bits of the Sequence Number of the packet whose full header is used as a reference to form RefSeqNum
350
Write delta_timestamp, reduced_pckt_id, reduced_SeqNum, RefSeqNum, and other fields into a reduced size header
399
END

FIG. 3

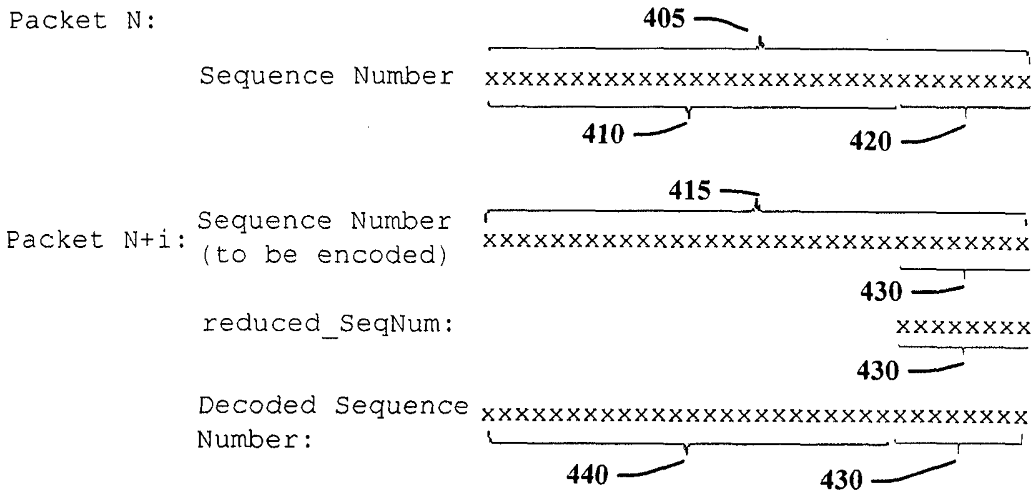

FIG. 4

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|C|I|Q|F|P|FEC| TB  | DS  |R|RES|           packet_id           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     packet_sequence_number                    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           timestamp                           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  TP |  flow_label |e| reserved|      private_user_data        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            payload data                  .... 
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 5A

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|C|Q|F|P|FEC| TB  | DS  |R|          delta_timestamp            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|reduced_pckt_id|reduced_SeqNum |RefSeqNum|  TP |  flow_label |e|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       private_user_data       |          payload data      ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 5B

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|C|I|f_i|A|R|P|E|             length              |     type    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  data_offset  |  frag_count  | numDU |       DU_offset        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| ...  |           DU_offset            |         ....          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                  payload_sequence_number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      header_extension                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        payload data                       ....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 6A

```
 0                   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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|C|f_i|E|     type      |  data_offset  |  frag_count   |RefSNum|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       header_extension                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         payload data                      ....
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

FIG. 6B

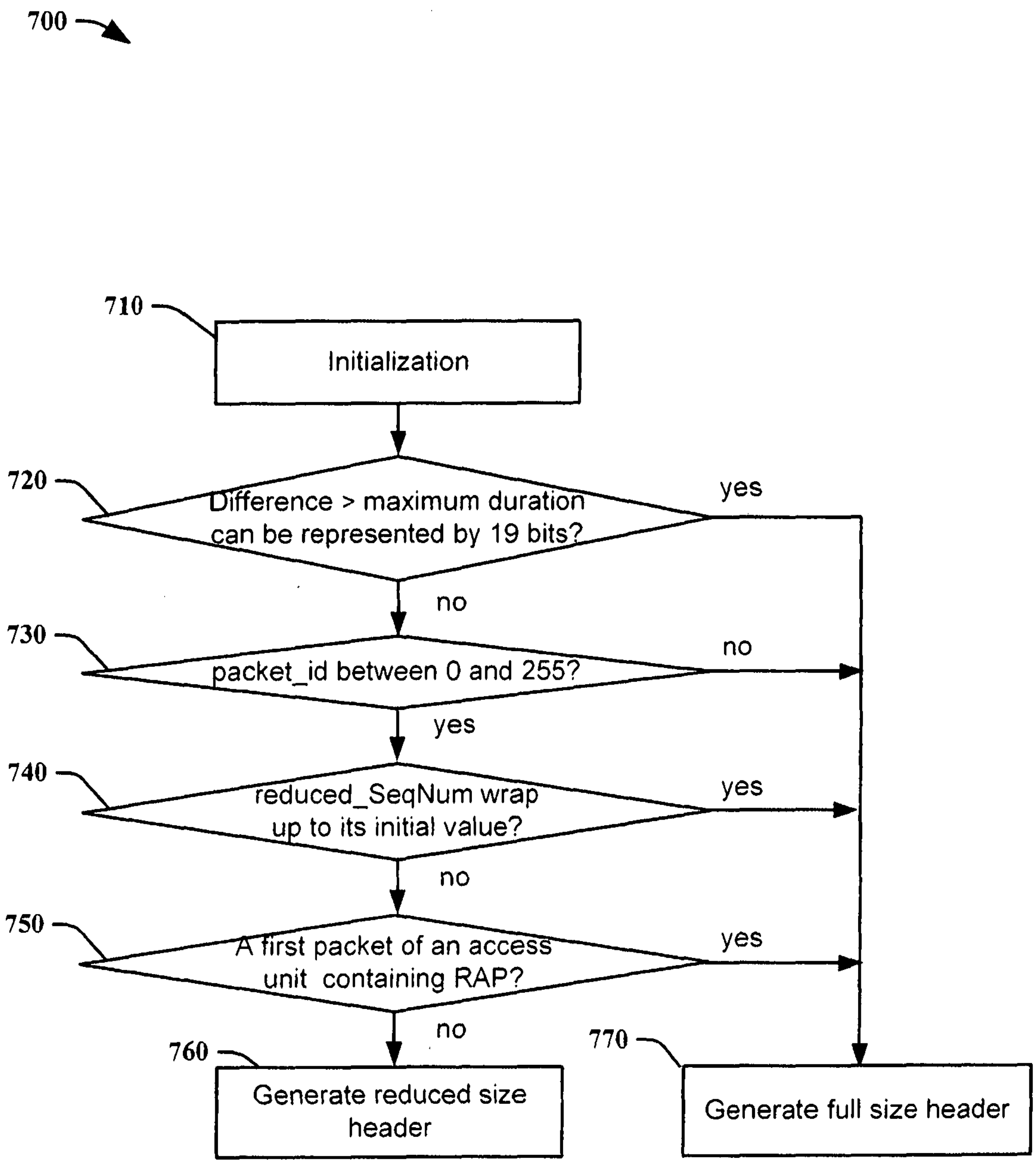
700
710
Initialization
720
Difference > maximum duration can be represented by 19 bits?
yes
no
730
packet_id between 0 and 255?
no
yes
740
reduced_SeqNum wrap up to its initial value?
yes
no
750
A first packet of an access unit containing RAP?
yes
no
760
Generate reduced size header
770
Generate full size header

FIG. 7

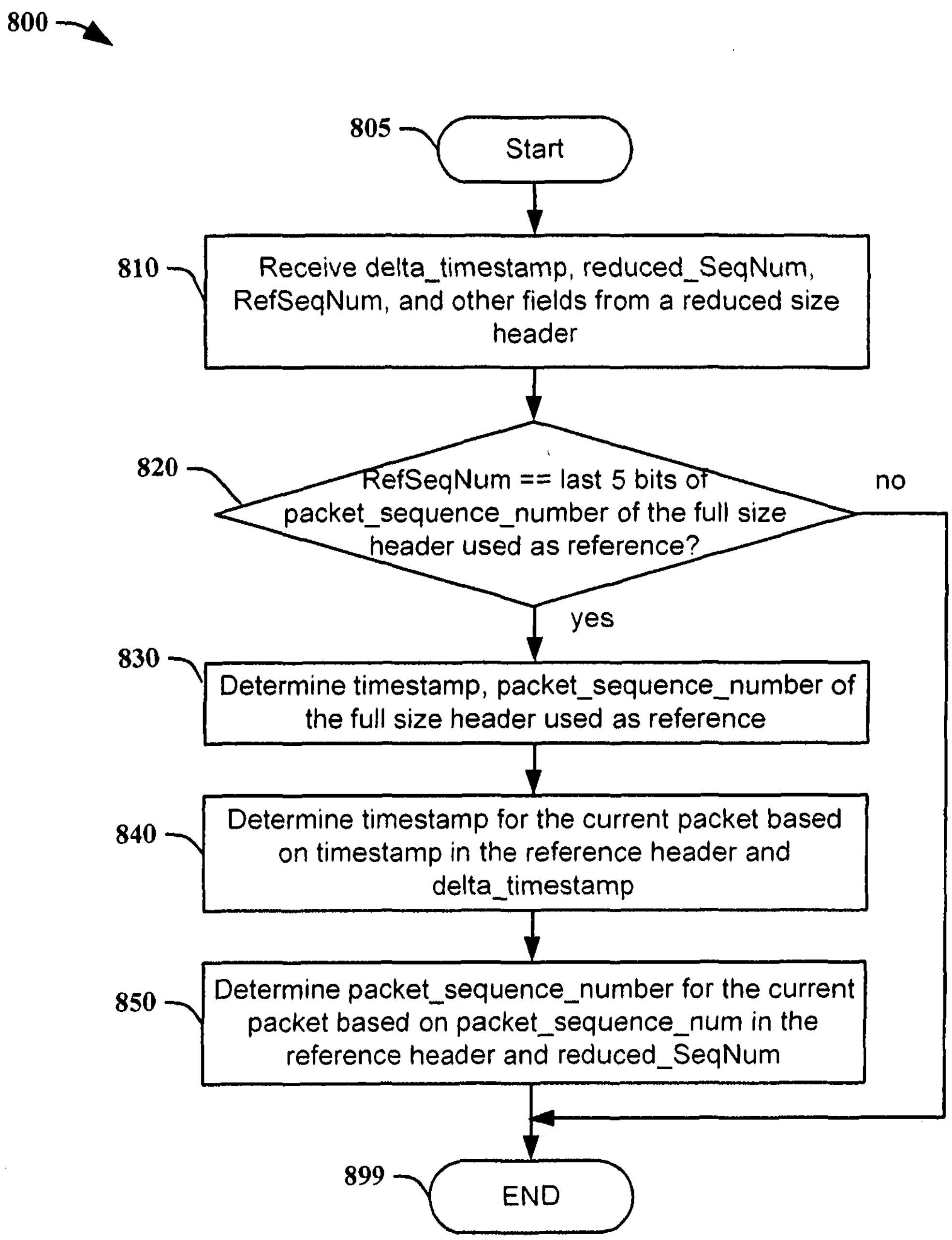
800
805
Start
810
Receive delta_timestamp, reduced_SeqNum, RefSeqNum, and other fields from a reduced size header
820
RefSeqNum == last 5 bits of packet_sequence_number of the full size header used as reference?
no
yes
830
Determine timestamp, packet_sequence_number of the full size header used as reference
840
Determine timestamp for the current packet based on timestamp in the reference header and delta_timestamp
850
Determine packet_sequence_number for the current packet based on packet_sequence_num in the reference header and reduced_SeqNum
899
END

FIG. 8

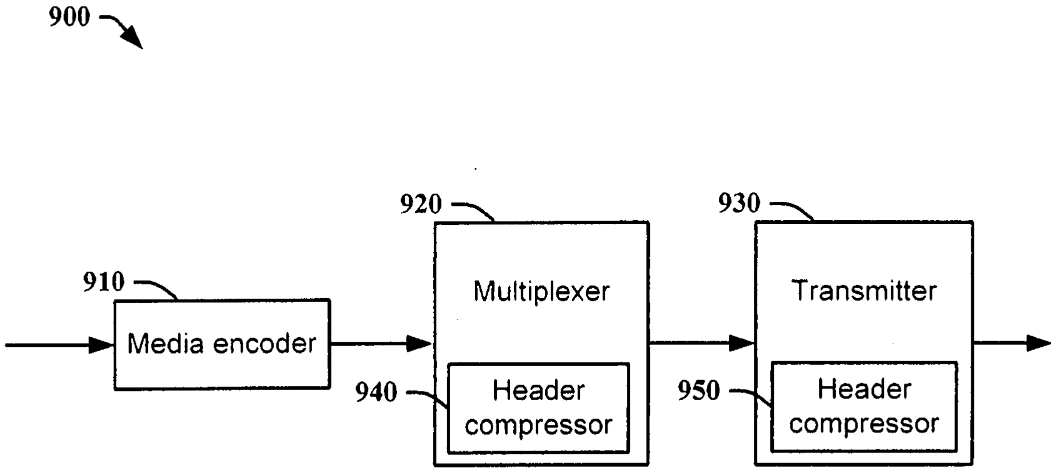
900
910
Media encoder
920
Multiplexer
940
Header compressor
930
Transmitter
950
Header compressor

FIG. 9

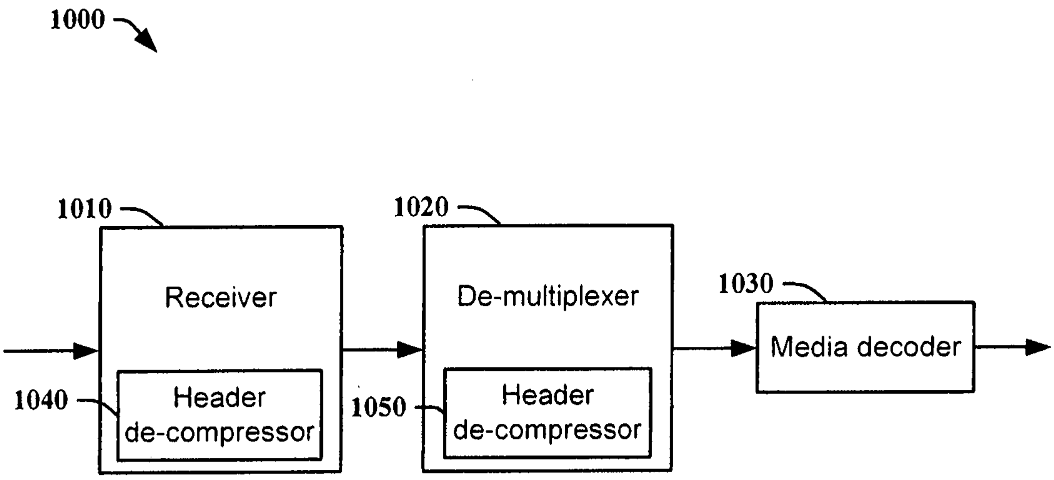
1000
1010
Receiver
1040
Header de-compressor
1020
De-multiplexer
1050
Header de-compressor
1030
Media decoder

FIG. 10

1100

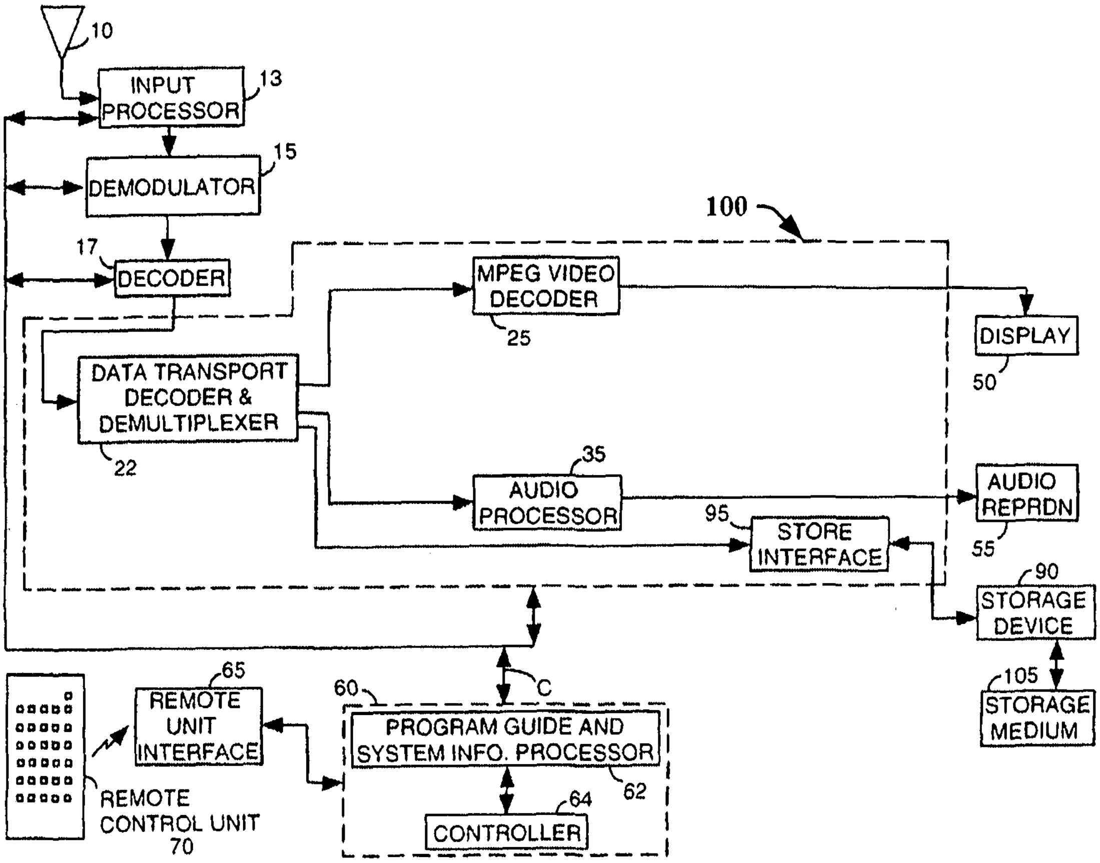
10
INPUT PROCESSOR
13
DEMODULATOR
15
17
DECODER
100
MPEG VIDEO DECODER
25
DISPLAY
50
DATA TRANSPORT DECODER & DEMULTIPLEXER
22
35
AUDIO PROCESSOR
AUDIO REPRDN
55
95
STORE INTERFACE
90
STORAGE DEVICE
105
STORAGE MEDIUM
65
REMOTE UNIT INTERFACE
60
C
PROGRAM GUIDE AND SYSTEM INFO. PROCESSOR
62
64
CONTROLLER
REMOTE CONTROL UNIT
70

FIG. 11

INTERNATIONAL SEARCH REPORT

International application No
PCT/IB2013/002615

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04L29/06
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

EPO-Internal, IBM-TDB, WPI Data

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2003/193950 A1 (PHILIPS DAVID [IL] ET AL) 16 October 2003 (2003-10-16) figure 2 paragraph [0003] paragraph [0027] - paragraph [0030] paragraph [0038] paragraph [0072] -----	1-16
X	WO 2007/112140 A2 (QUALCOMM INC [US]; KAPOOR ROHIT [US]; KRETZ MAGNUS [US]) 4 October 2007 (2007-10-04) paragraph [0034] - paragraph [0037] -----	1-16
A	US 2012/201205 A1 (GOPALAKRISHNAN NANDU [US] ET AL) 9 August 2012 (2012-08-09) figure 3a figure 3b paragraph [0011] -----	1-16

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 February 2014	03/03/2014

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Suciu, Radu

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/IB2013/002615

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003193950 A1	16-10-2003	AT 488942 T	15-12-2010
		AU 2003219488 A1	27-10-2003
		EP 1495612 A1	12-01-2005
		IL 149165 A	10-12-2006
		US 2003193950 A1	16-10-2003
		WO 03088614 A1	23-10-2003
WO 2007112140 A2	04-10-2007	AU 2007230862 A1	04-10-2007
		BR PI0706417 A2	29-03-2011
		CA 2633896 A1	04-10-2007
		CN 101366261 A	11-02-2009
		EP 1974528 A2	01-10-2008
		JP 5134115 B2	30-01-2013
		JP 2009522954 A	11-06-2009
		JP 2011239432 A	24-11-2011
		JP 2013243690 A	05-12-2013
		KR 20080083355 A	17-09-2008
		TW 200814667 A	16-03-2008
		US 2010278196 A1	04-11-2010
		WO 2007112140 A2	04-10-2007
US 2012201205 A1	09-08-2012	CN 103348620 A	09-10-2013
		EP 2673925 A1	18-12-2013
		US 2012201205 A1	09-08-2012
		WO 2012108989 A1	16-08-2012

摘要

头部可以被区别编码以有效地压缩头部。在一个实施例中，当前字段和参考字段之间的差异可以被用于缩减的头部中。在另一实施例中，当前字段的最低有效位可以被用于表示当前字段。当接收到此类字段时，需要参考字段的最高有效位来对该字段进行解压缩。此外，本原理确定了可以从头部压缩中受益的典型使用情境，并且进一步确定了在缩减的头部中可以移除的一些字段或者可以由更少的位来表示的一些字段。头部压缩可以强加限制于当前字段的表示。本原理承认使用缩减的头部的限制并且提供了关于应用头部压缩的规则和指导。