



(86) Date de dépôt PCT/PCT Filing Date: 2012/05/25  
(87) Date publication PCT/PCT Publication Date: 2012/11/29  
(45) Date de délivrance/Issue Date: 2016/12/06  
(85) Entrée phase nationale/National Entry: 2013/11/21  
(86) N° demande PCT/PCT Application No.: KR 2012/004167  
(87) N° publication PCT/PCT Publication No.: 2012/161552  
(30) Priorités/Priorities: 2011/05/25 (US61/489,683);  
2011/06/06 (US61/493,964); 2011/07/07 (US61/505,512)

(51) Cl.Int./Int.Cl. *H04N 7/015* (2006.01),  
*H04H 20/59* (2009.01), *H04W 4/06* (2009.01)  
(72) Inventeurs/Inventors:  
KWAK, MINSUNG, KR;  
KIM, JEONGWOO, KR  
(73) Propriétaire/Owner:  
LG ELECTRONICS INC., KR  
(74) Agent: SMART & BIGGAR

(54) Titre : SYSTEME DE TRANSMISSION/DE RECEPTION ET PROCEDE PERMETTANT DE TRAITER UN SIGNAL DE  
DIFFUSION

(54) Title: TRANSMITTING/RECEIVING SYSTEM AND METHOD FOR PROCESSING A BROADCASTING SIGNAL

| Syntax                                                                                                                                                                                             | Bits                            | Format                                                           |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|------------------------------------------------------------------|
| FIC -Segment_header () {<br>FIC_segment_type<br>wake_up_indicator<br>FIC_chunk_major_protocol_version<br>current_next_indicator<br>error_indicator<br>FIC_segment_num<br>FIC_last_segment_num<br>} | 2<br>2<br>2<br>1<br>1<br>4<br>4 | '00'<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf |

(57) **Abrégé/Abstract:**

According to the present invention, a method for processing a broadcast signal in a transmitting system that transmits an emergency alert message through a mobile broadcast network comprises the steps of: RS-CRC encoding of mobile service data containing an emergency alert message so as to generate an RS frame that belongs to an ensemble; dividing the RS frame into a plurality of RS frame portions; mapping the plurality of RS frame portions to data groups and inserting one FIC segment, TPC data, and a plurality of base data streams into each data group; trellis-encoding the data of the data groups; and transmitting a broadcast signal comprising the trellis-encoded data. At least one of either a FIC segment header contained in the FIC segment or the TPC data includes wake-up indication information for compulsorily placing a broadcast receiver into an active mode depending on the level of seriousness of the emergency alert message.

## (12) 특허협력조약에 의하여 공개된 국제출원

(19) 세계지식재산권기구  
국제사무국(43) 국제공개일  
2012년 11월 29일 (29.11.2012)(10) 국제공개번호  
**WO 2012/161552 A3**

- (51) 국제특허분류:  
H04N 7/015 (2006.01) H04W 4/06 (2009.01)  
H04H 20/59 (2008.01)
- (21) 국제출원번호: PCT/KR2012/004167
- (22) 국제출원일: 2012년 5월 25일 (25.05.2012)
- (25) 출원언어: 한국어
- (26) 공개언어: 한국어
- (30) 우선권정보:  
61/489,683 2011년 5월 25일 (25.05.2011) US  
61/493,964 2011년 6월 6일 (06.06.2011) US  
61/505,512 2011년 7월 7일 (07.07.2011) US
- (71) 출원인 (US을(를) 제외한 모든 지정국에 대하여): **엘지전자 주식회사 (LG ELECTRONICS INC.)** [KR/KR]; 150-721 서울 영등포구 여의도동 20, Seoul (KR).
- (72) 발명자; 겸
- (75) 발명자/출원인 (US에 한하여): **곽민성 (KWAK, Min-sung)** [KR/KR]; 137-724 서울시 서초구 우면동 16번지 엘지전자 특허센터, Seoul (KR). **김정우 (KIM, Jeong-woo)** [KR/KR]; 137-724 서울시 서초구 우면동 16번지 엘지전자 특허센터, Seoul (KR).
- (74) 대리인: **김용인 (KIM, Yong In)** 등; 138-861 서울 송파구 잠실동 175-9 현대빌딩 7층 KBK 특허법률사무소, Seoul (KR).

(81) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 국내 권리의 보호를 위하여): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, 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, IS, JP, KE, KG, KM, 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, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 역내 권리의 보호를 위하여): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, SZ, TZ, UG, ZM, ZW), 유라시아 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 유럽 (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, ML, MR, NE, SN, TD, TG).

공개:

— 국제조사보고서와 함께 (조약 제 21 조(3))

(88) 국제조사보고서 공개일: 2013년 3월 21일

(54) Title: TRANSMITTING/RECEIVING SYSTEM AND METHOD FOR PROCESSING A BROADCAST SIGNAL

(54) 발명의 명칭 : 송/수신 시스템 및 방송 신호 처리 방법

[Fig. 40]

| Syntax                           | Bits | Format |
|----------------------------------|------|--------|
| FIC-Segment_header() {           |      |        |
| FIC_segment_type                 | 2    | '00'   |
| wake_up_indicator                | 2    | uimsbf |
| FIC_chunk_major_protocol_version | 2    | uimsbf |
| current_next_indicator           | 1    | uimsbf |
| error_indicator                  | 1    | uimsbf |
| FIC_segment_num                  | 4    | uimsbf |
| FIC_last_segment_num             | 4    | uimsbf |
| }                                |      |        |

(57) Abstract: According to the present invention, a method for processing a broadcast signal in a transmitting system that transmits an emergency alert message through a mobile broadcast network comprises the steps of: RS-CRC encoding of mobile service data containing an emergency alert message so as to generate an RS frame that belongs to an ensemble; dividing the RS frame into a plurality of RS frame portions; mapping the plurality of RS frame portions to data groups and inserting one FIC segment, TPC data, and a plurality of base data streams into each data group; trellis-encoding the data of the data groups; and transmitting a broadcast signal comprising the trellis-encoded data. At least one of either a FIC segment header contained in the FIC segment or the TPC data includes wake-up indication information for compulsorily placing a broadcast receiver into an active mode depending on the level of seriousness of the emergency alert message.

(57) 요약서:

[다음 쪽 계속]

WO 2012/161552 A3

**WO 2012/161552 A3**

---

모바일 방송망을 통해 긴급 경고 메시지를 전송하는 송신 시스템의 방송 신호 처리 방법은 긴급 경고 메시지를 포함하는 모바일 서비스 데이터에 대해 RS-CRC 인코딩을 수행하여 앙상블에 속하는 RS 프레임을 생성하는 단계, 상기 RS 프레임을 복수개의 RS 프레임 포션들로 구분하는 단계, 상기 복수개의 RS 프레임 포션들을 각 데이터 그룹에 매핑하고, 하나의 FIC 세그먼트, TPC 데이터 및 복수개의 기지 데이터 열들을 각 데이터 그룹에 삽입하는 단계, 상기 데이터 그룹들의 데이터에 대해 트렐리스 인코딩을 수행하는 단계, 및 상기 트렐리스 인코딩된 데이터를 포함하는 방송 신호를 전송하는 단계를 포함하며, 상기 FIC 세그먼트에 포함된 FIC 세그먼트 헤더와 상기 TPC 데이터 중 적어도 하나는 상기 긴급 경고 메시지의 심각한 정도에 따라 방송 수신기를 액티브 모드로 강제 전환하기 위한 웨이크-업 지시 정보를 포함한다.

**[DESCRIPTION]****[Invention Title]**

TRANSMITTING/RECEIVING SYSTEM AND METHOD FOR PROCESSING A BROADCASTING SIGNAL

**[Technical Field]**

**[0001]** The present invention relates to a digital broadcasting system for transmitting and receiving digital broadcast signal, and more particularly, to a transmitting system for processing and transmitting digital broadcast signal, and a receiving system for receiving and processing digital broadcast signal and, a method of processing digital broadcast signal in the transmitting system and the receiving system.

**[Background Art]**

**[0002]** The Vestigial Sideband (VSB) transmission mode, which is adopted as the standard for digital broadcasting in North America and the Republic of Korea, is a system using a single carrier method. Therefore, the receiving performance of the digital broadcast receiving system may be deteriorated in a poor channel environment. Particularly, since resistance to changes in channels and noise is more highly required when using portable and/or mobile broadcast receivers, the receiving performance may be even more deteriorated when transmitting mobile service data by the VSB transmission mode.

**[Disclosure]****[Technical Problem]**

**[0003]** An object of the present invention is to provide a transmitting system and a receiving system and a method of processing broadcast signal that are highly resistant to channel changes and noise.

**[0004]** Another object of the present invention is to provide a transmitting system and a receiving system and a method of processing broadcast signal that can enhance the receiving performance of the receiving system by performing additional

encoding on mobile service data and by transmitting the processed data to the receiving system.

[0005] A further object of the present invention is to provide a transmitting system and a receiving system and a method of processing broadcast signal that can also enhance the receiving performance of the receiving system by inserting known data already known in accordance with a pre-agreement between the receiving system and the transmitting system in a predetermined region within a data region.

[0006] Yet another object of the present invention is to provide a transmitting system, a receiving system, and a method of processing a broadcast signal that may allow the transmitting system to signal and transmit information identifying an enhancement service, and that may then allow the receiving system to use the transmitted information so as to support to the enhancement service.

[0007] Another object of the present invention is to provide a transmitting system and a receiving system and a method of processing broadcast signals that allow the receiving system capable of receiving mobile broadcasts to provide a disaster alert service by transmitting an emergency alert message using a mobile broadcast network.

[0008] Another object of the present invention is to provide a signaling method and transmission/reception method for signaling information that allow the receiving system capable of receiving mobile broadcast to provide the disaster alert service.

[0009] Another object of the present invention is to provide a transmitting system and a receiving system and a method of processing broadcast signals which can efficiently provide a disaster alert service by providing additional information associated with the emergency alert message.

**[Technical Solution]**

[0010] The object of the present invention can be achieved by

providing a method for processing a broadcast signal in a transmitting system including the steps of performing Reed Solomon-Cyclic Redundancy Check (RS-CRC) encoding on mobile service data containing an emergency alert message and generating an RS frame belonging to an ensemble, dividing the RS frame into a plurality of RS frame portions, mapping the RS frame portions into respective data groups and inserting one fast information channel (FIC) segment and a plurality of known data sequences into each of the data groups, performing trellis encoding on data of the data groups, and transmitting the broadcast signal including the trellis-encoded data.

**[0011]** An FIC chunk is configured with an FIC chunk header and FIC chunk payload, signals bidding information between the ensemble and a mobile service included in the ensemble, and is divided into a plurality of FIC segment payloads. The FIC segment is configured with an FIC segment header and one of the plurality of FIC segment payloads.

**[0012]** The FIC segment header includes wake-up indication information for compulsorily switching a broadcast receiver to an active mode according to a degree of severity of the emergency alert message.

**[0013]** The emergency alert message may be transmitted in a Common Alerting Protocol (CAP) form or a syntax form.

**[0014]** The ensemble may include at least one of a service map table (SMT) and an emergency alert table (EAT). Each of the SMT and the EAT comprises an ensemble identifier to identify the ensemble. Identifying information for identifying whether the emergency alert message is in the CAP form or in the syntax form may be signaled to at least one of the SMT and the EAT.

**[0015]** IP datagrams including the SMT and IP datagrams including the EAT may be transmitted over a service signaling channel. All the IP datagrams transmitted over the service signaling channel may have the same well-known IP address and the same

74420-663

well-known UDP port number.

[0016] The FIC chunk payload may include the ensemble identifier to identify the ensemble and indication information to indicate whether the EAT is transmitted over a service signaling channel included in the ensemble.

[0017] The method may further include the step of including disaster-related additional information associated with the emergency alert message in at least one file and transmitting the at least file including the disaster-related additional information in non-real time (NRT).

[0018] In another aspect of the present invention, provided herein is a transmitting system including a first encoder configured to perform Reed Solomon-Cyclic Redundancy Check (RS-CRC) encoding on mobile service data containing an emergency alert message and generate an RS frame belonging to an ensemble, a divider configured to divide the RS frame into a plurality of RS frame portions, a group formatter configured to map the RS frame portions into respective data groups and insert one fast information channel (FIC) segment and a plurality of known data sequences into each of the data groups, a second encoder configured to perform trellis encoding on data of the data groups, and a transmitting unit configured to transmit a broadcast signal including the trellis-encoded data.

[0019] An FIC chunk is configured with an FIC chunk header and FIC chunk payload, signals bidding information between the ensemble and a mobile service included in the ensemble, and is divided into a plurality of FIC segment payloads. The FIC segment may be configured with an FIC segment header and one of the plurality of FIC segment payloads. The FIC segment header may include wake-up indication information for compulsorily switching a broadcast receiver to an active mode according to a degree of severity of the emergency alert message.

74420-663

**[0019a]** According to another aspect of the present disclosure, there is provided a method for processing a broadcast signal in a transmitting system, the method comprising: performing Reed Solomon-Cyclic Redundancy Check (RS-CRC) encoding on mobile service data, thereby generating an RS frame belonging to an ensemble; dividing the RS frame into a plurality of RS frame portions; mapping the RS frame portions into each of data groups and inserting one fast information channel (FIC) segment and a plurality of known data sequences into each of the data groups; performing trellis encoding on data of the data groups; and transmitting the broadcast signal including the trellis-encoded data, wherein an FIC chunk is configured with an FIC chunk header and FIC chunk payload, signals binding information between the ensemble and a mobile service included in the ensemble, and is divided into a plurality of FIC segment payloads, wherein the FIC segment is configured with an FIC segment header and one of the plurality of FIC segment payloads, wherein the FIC segment header comprises wake-up indication information which is used to determine at a receiver whether the receiver should wake up or not, and wherein an emergency alert message is transferred through at least an emergency alert table (EAT) or an IP datagram.

**[0019b]** There is also provided a transmitting system comprising: a first encoder configured to perform Reed Solomon-Cyclic Redundancy Check (RS-CRC) encoding on mobile service data, thereby generating an RS frame belonging to an ensemble; a divider configured to divide the RS frame into a plurality of RS frame portions; a group formatter configured to map the RS frame portions into each of data groups and insert

74420-663

one fast information channel (FIC) segment and a plurality of known data sequences into each of the data groups; a second encoder configured to perform trellis encoding on data of the data groups; and a transmitting unit configured to transmit a broadcast signal including the trellis-encoded data, wherein an FIC chunk is configured with an FIC chunk header and FIC chunk payload, signals binding information between the ensemble and a mobile service included in the ensemble, and is divided into a plurality of FIC segment payloads, wherein the FIC segment is configured with an FIC segment header and one of the plurality of FIC segment payloads, and wherein the FIC segment header comprises wake-up indication information which is used to determine at a receiver whether the receiver should wake up or not, and wherein an emergency alert message is transferred through at least an emergency alert table (EAT) or an IP datagram.

**[0020]** Other objects, features and advantages of the invention

will be apparent from a detailed description of embodiments with reference to the accompanying drawings.

**[Advantageous Effects]**

[0021] As described above, the transmitting system and the receiving system and the broadcast signal processing method of the same according to the present invention have the following advantages. When transmitting mobile service data through a channel, the present invention may be robust against errors and backward compatible with the conventional digital broadcast receiving system.

[0022] Moreover, the present invention may also receive the mobile service data without any error even in channels having severe ghost effect and noise.

[0023] Furthermore, by inserting known data in a particular position (or place) within a data region and transmitting the processed data, the receiving performance of the receiving system may be enhanced even in a channel environment that is liable to frequent changes.

[0024] In addition, by transmitting an emergency alert message using a mobile broadcast network in a mobile broadcast transmitter and, receiving and processing it by a mobile broadcast receiver, the broadcast receiver capable of receiving mobile broadcasts can provide a disaster alert service to a viewer.

[0025] Also, by transmitting and receiving additional information associated with the emergency alert message in non-real time and then by servicing to viewers, the disaster alert service can be serviced efficiently through mobile broadcast.

[0026] Finally, the present invention is even more effective when applied to mobile and portable receivers, which are also liable to a frequent change in channel and which require protection (or resistance) against intense noise.

**[Description of Drawings]**

- [0027] FIG. 1 illustrates a structure of a M/H frame for transmitting and receiving mobile service data according to the present invention;
- [0028] FIG. 2 illustrates an exemplary structure of a VSB frame;
- [0029] FIG. 3 illustrates a mapping example of the positions to which the first 4 slots of a sub-frame are assigned with respect to a VSB frame in a space region;
- [0030] FIG. 4 illustrates a mapping example of the positions to which the first 4 slots of a sub-frame are assigned with respect to a VSB frame in a time region;
- [0031] FIG. 5 illustrates an alignment of data after being data interleaved and identified;
- [0032] FIG. 6 illustrates an enlarged portion of the data group shown in FIG. 5 for a better understanding of the present invention;
- [0033] FIG. 7 illustrates an alignment of data before being data interleaved and identified;
- [0034] FIG. 8 illustrates an enlarged portion of the data group shown in FIG. 7 for a better understanding of the present invention;
- [0035] FIG. 9 illustrates an exemplary assignment order of data groups being assigned to one of 5 sub-frames according to the present invention;
- [0036] FIG. 10 illustrates an example of assigning a single parade to an M/H frame according to the present invention;
- [0037] FIG. 11 illustrates an example of RS frame payload according to the present invention;
- [0038] FIG. 12 is a diagram illustrating a structure of an M/H header within an M/H service data packet according to the present invention;
- [0039] FIG. 13(a) and FIG. 13(b) are diagrams illustrating another example of RS frame payload according to the present invention; and

- [0040] FIG. 14 illustrates a block diagram showing an embodiment of a transmitter according to the present invention;
- [0041] FIG. 15 illustrates a block diagram showing an example of a pre-processor according to the present invention;
- [0042] FIG. 16 illustrates a conceptual block diagram of the M/H frame encoder according to the present invention;
- [0043] FIG. 17 illustrates a detailed block diagram of an RS frame encoder according to the present invention;
- [0044] FIG. 18(a) and FIG. 18(b) illustrate a process of one or two RS frame being divided into several portions, based upon an RS frame mode value, and a process of each portion being assigned to a corresponding region within the respective data group;
- [0045] FIG. 19(a) to FIG. 19(c) illustrate error correction encoding and error detection encoding processes according to an embodiment of the present invention;
- [0046] FIG. 20(a) and FIG. 20(b) illustrate an example which a parade configures of two RS frames
- [0047] FIG. 21(a) and FIG. 21(b) illustrate an exemplary process of dividing an RS frame for configuring a data group according to the present invention;
- [0048] FIG. 22 illustrates a block diagram of a block processor according to an embodiment of the present invention;
- [0049] FIG. 23 illustrates a detailed block diagram of a convolution encoder of the block processor;
- [0050] FIG. 24 illustrates a symbol interleaver of the block processor;
- [0051] FIG. 25 illustrates a block diagram of a group formatter according to an embodiment of the present invention;
- [0052] FIG. 26 illustrates a block diagram of a trellis encoder according to an embodiment of the present invention;

- [0053] FIG. 27 illustrates an example of assignment of training sequences within a data group before trellis encoding according to the present invention;
- [0054] FIG. 28 illustrates an example of assignment of training sequences within a data group after trellis encoding according to the present invention; FIG. 29 illustrates an example of assigning signaling information area according to an embodiment of the present invention;
- [0055] FIG. 30 illustrates a detailed block diagram of a signaling encoder according to the present invention;
- [0056] FIG. 31 illustrates an example of a syntax structure of TPC data according to the present invention;
- [0057] FIG. 32 illustrates an example of a transmission scenario of the TPC data and the FIC data level according to the present invention;
- [0058] FIG. 33 illustrates a syntax structure of an FIC chunk according to an embodiment of the present invention;
- [0059] FIG. 34 illustrates a syntax structure of an FIC chunk header according to an embodiment of the present invention;
- [0060] FIG. 35 illustrates a syntax structure of an FIC chunk payload according to an embodiment of the present invention;
- [0061] FIG. 36 illustrates a syntax structure of an FIC segment header according to an embodiment of the present invention;
- [0062] FIG. 37 illustrates a syntax structure of a service map table (SMT) according to an embodiment of the present invention;
- [0063] FIG. 38 illustrates a bitstream syntax structure of component\_descriptor() according to an embodiment of the present invention;

- [0064] FIG. 39 illustrates a syntax structure of an FIC segment header according to an embodiment of the present invention;
- [0065] FIG. 40 illustrates a syntax structure of an FIC segment header according to another embodiment of the present invention;
- [0066] FIG. 41 illustrates a table showing meaning of values of a wake-up indicator field according to an embodiment of the present invention;
- [0067] FIG. 42 illustrates a syntax structure of an FIC chunk payload according to another embodiment of the present invention;
- [0068] FIG. 43 illustrates a syntax structure of an EAS message descriptor according to an embodiment of the present invention;
- [0069] FIG. 44 illustrates a syntax structure of an EAS message descriptor according to another embodiment of the present invention;
- [0070] FIG. 45 illustrates an example of signaling an EAS message descriptor to ensemble level according to an embodiment of the present invention;
- [0071] FIG. 46 illustrates an example of signaling an EAS message descriptor to service level according to an embodiment of the present invention;
- [0072] FIG. 47 illustrates an example of displaying an emergency alert message on a portion of screen according to an embodiment of the present invention;
- [0073] FIG. 48 illustrates an example of displaying additional information related to an emergency alert message on another portion of screen according to an embodiment of the present invention;
- [0074] FIG. 49 illustrates a syntax structure of capabilities descriptor according to the present invention;

- [0075] FIG. 50 illustrates a syntax structure of NRT service descriptor according to the present invention;
- [0076] FIG. 51 illustrates a table showing meaning of values of a consumption model field according to an embodiment of the present invention;
- [0077] FIG. 52 illustrates a flow chart for receiving and displaying disaster information and disaster related-additional information by a broadcast receiver according to an embodiment of the present invention;
- [0078] FIG. 53 illustrates a flow chart for receiving and displaying disaster information and disaster related-additional information by a broadcast receiver according to another embodiment of the present invention;
- [0079] FIG. 54 illustrates an example of encapsulating an emergency alert message to IP datagram according to the present invention;
- [0080] FIG. 55 illustrates an example of a syntax structure of emergency alert system descriptor capable of transmitting in a payload of IP datagram of FIG. 54;
- [0081] FIG. 56 illustrates a table showing meaning of values of a service category field of SMT according to an embodiment of the present invention;
- [0082] FIG. 57 illustrates an example of a bitstream syntax structure of component\_data() providing signaling information for emergency alert message according to an embodiment of the present invention;
- [0083] FIG. 58 illustrates an example of a bitstream syntax structure of EAT section according to an embodiment of the present invention;
- [0084] FIG. 59 illustrates a flow chart for receiving and displaying disaster information and disaster related-additional information by a broadcast receiver according to another embodiment of the present invention; and

[0085] FIG. 60 illustrates a block diagram of a receiving system according to an embodiment of the present invention.

[Best Mode]

[0086] Reference will now be made in detail to the preferred embodiments of the present invention, examples of which are illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

[0087]

[0088] Definition of terms used in the present invention

[0089] In addition, although the terms used in the present invention are selected from generally known and used terms, some of the terms mentioned in the description of the present invention have been selected by the applicant at his or her discretion, the detailed meanings of which are described in relevant parts of the description herein. Furthermore, it is required that the present invention is understood, not simply by the actual terms used but by the meaning of each term lying within.

[0090] Among the terms used in the description of the present invention, main service data correspond to data that can be received by a fixed receiving system and may include audio/video (A/V) data. More specifically, the main service data may include A/V data of high definition (HD) or standard definition (SD) levels and may also include diverse data types required for data broadcasting. Also, the known data correspond to data pre-known in accordance with a pre-arranged agreement between the receiving system and the transmitting system.

[0091] Additionally, among the terms used in the present invention, "M/H (or MH)" corresponds to the initials of "mobile" and "handheld" and represents the opposite concept of a fixed-type system. Furthermore, the M/H service data may include at least one of mobile service data and handheld

service data, and will also be referred to as "mobile service data" for simplicity. Herein, the mobile service data not only correspond to M/H service data but may also include any type of service data with mobile or portable characteristics. Therefore, the mobile service data according to the present invention are not limited only to the M/H service data.

[0092] The above-described mobile service data may correspond to data having information, such as program execution files, stock information, and so on, and may also correspond to A/V data. Most particularly, the mobile service data may correspond to A/V data having lower resolution and lower data rate as compared to the main service data. For example, if an A/V codec that is used for a conventional main service corresponds to a MPEG-2 codec, a MPEG-4 advanced video coding (AVC) or scalable video coding (SVC) having better image compression efficiency may be used as the A/V codec for the mobile service. Furthermore, any type of data may be transmitted as the mobile service data. For example, transport protocol expert group (TPEG) data for broadcasting real-time transportation information may be transmitted as the main service data.

[0093] Also, a data service using the mobile service data may include weather forecast services, traffic information services, stock information services, viewer participation quiz programs, real-time polls and surveys, interactive education broadcast programs, gaming services, services providing information on synopsis, character, background music, and filming sites of soap operas or series, services providing information on past match scores and player profiles and achievements, and services providing information on product information and programs classified by service, medium, time, and theme enabling purchase orders to be processed. Herein, the present invention is not limited only to the services mentioned above.

[0094] In the present invention, the transmitting system provides backward compatibility in the main service data so as to be received by the conventional receiving system. Herein, the main service data and the mobile service data are multiplexed to the same physical channel and then transmitted.

[0095] Furthermore, the transmitting system according to the present invention performs additional encoding on the mobile service data and inserts the data already known by the receiving system and transmitting system (e.g., known data), thereby transmitting the processed data.

[0096] Therefore, when using the transmitting system according to the present invention, the receiving system may receive the mobile service data during a mobile state and may also receive the mobile service data with stability despite various distortion and noise occurring within the channel.

[0097]

[0098] M/H Frame Structure

[0099] In the embodiment of the present invention, the mobile service data are first multiplexed with main service data in M/H frame units and, then, modulated in a VSB mode and transmitted to the receiving system.

[00100] At this point, one M/H frame configures of K1 number of sub-frames, wherein one sub-frame includes K2 number of slots. Also, each slot may be configured of K3 number of data packets. In the embodiment of the present invention, K1 will be set to 5, K2 will be set to 16, and K3 will be set to 156 (i.e., K1=5, K2=16, and K3=156). The values for K1, K2, and K3 presented in this embodiment either correspond to values according to a preferred embodiment or are merely exemplary. Therefore, the above-mentioned values will not limit the scope of the present invention.

[00101] FIG. 1 illustrates a structure of an M/H frame for transmitting and receiving mobile service data according to the present invention. In the example shown in FIG. 1, one M/H

frame consists of 5 sub-frames, wherein each sub-frame includes 16 slots. In this case, the M/H frame according to the present invention includes 5 sub-frames and 80 slots.

[00102] Also, in a packet level, one slot is configured of 156 data packets (*i.e.*, transport stream packets), and in a symbol level, one slot is configured of 156 data segments. Herein, the size of one slot corresponds to one half ( $1/2$ ) of a VSB field. More specifically, since one 207-byte data packet has the same amount of data as a data segment, a data packet prior to being interleaved may also be used as a data segment.

[00103] At this point, two VSB fields are grouped to form a VSB frame.

[00104] FIG. 2 illustrates an exemplary structure of a VSB frame, wherein one VSB frame consists of 2 VSB fields (*i.e.*, an odd field and an even field). Herein, each VSB field includes a field synchronization segment and 312 data segments.

[00105] The slot corresponds to a basic time period for multiplexing the mobile service data and the main service data. Herein, one slot may either include the mobile service data or be configured only of the main service data.

[00106] If one M/H frame is transmitted during one slot, the first 118 data packets within the slot correspond to a data group. And, the remaining 38 data packets become the main service data packets. In another example, when no data group exists in a slot, the corresponding slot is configured of 156 main service data packets.

[00107] Meanwhile, when the slots are assigned to a VSB frame, an offset exists for each assigned position.

[00108] FIG. 3 illustrates a mapping example of the positions to which the first 4 slots of a sub-frame are assigned with respect to a VSB frame in a space region. And, FIG. 4 illustrates a mapping example of the positions to which the first 4 slots of a sub-frame are assigned with respect to a VSB frame in a time region.

**[00109]** Referring to FIG. 3 and FIG. 4, a 38<sup>th</sup> data packet (TS packet #37) of a 1<sup>st</sup> slot (Slot #0) is mapped to the 1<sup>st</sup> data packet of an odd VSB field. A 38<sup>th</sup> data packet (TS packet #37) of a 2<sup>nd</sup> slot (Slot #1) is mapped to the 157<sup>th</sup> data packet of an odd VSB field. Also, a 38<sup>th</sup> data packet (TS packet #37) of a 3<sup>rd</sup> slot (Slot #2) is mapped to the 1<sup>st</sup> data packet of an even VSB field. And, a 38<sup>th</sup> data packet (TS packet #37) of a 4<sup>th</sup> slot (Slot #3) is mapped to the 157<sup>th</sup> data packet of an even VSB field. Similarly, the remaining 12 slots within the corresponding sub-frame are mapped in the subsequent VSB frames using the same method.

**[00110]** Meanwhile, one data group may be divided into at least one or more hierarchical regions. And, depending upon the characteristics of each hierarchical region, the type of mobile service data being inserted in each region may vary. For example, the data group within each region may be divided (or categorized) based upon the receiving performance.

**[00111]** In an example given in the present invention, a data group is divided into regions A, B, C, and D in a data configuration after data interleaving.

**[00112]** FIG. 5 illustrates an alignment of data after being data interleaved and identified. FIG. 6 illustrates an enlarged portion of the data group shown in FIG. 5 for a better understanding of the present invention. FIG. 7 illustrates an alignment of data before being data interleaved and identified. And, FIG. 8 illustrates an enlarged portion of the data group shown in FIG. 7 for a better understanding of the present invention. More specifically, a data structure identical to that shown in FIG. 5 is transmitted to a receiving system. In other words, one data packet is data-interleaved so as to be scattered to a plurality of data segments, thereby being transmitted to the receiving system. FIG. 5 illustrates an example of one data group being scattered to 170 data segments. At this point, since one 207-

byte packet has the same amount of data as one data segment, the packet that is not yet processed with data-interleaving may be used as the data segment.

**[00113]** FIG. 5 shows an example of dividing a data group prior to being data-interleaved into 10 M/H blocks (*i.e.*, M/H block 1 (B1) to M/H block 10 (B10)). In this example, each M/H block has the length of 16 segments. Referring to FIG. 5, only the RS parity data are allocated to a portion of 5 segments before the M/H block 1 (B1) and 5 segments behind the M/H block 10 (B10). The RS parity data are excluded in regions A to D of the data group.

**[00114]** More specifically, when it is assumed that one data group is divided into regions A, B, C, and D, each M/H block may be included in any one of region A to region D depending upon the characteristic of each M/H block within the data group. At this point, according to an embodiment of the present invention, each M/H block may be included in any one of region A to region D based upon an interference level of main service data.

**[00115]** Herein, the data group is divided into a plurality of regions to be used for different purposes. More specifically, a region of the main service data having no interference or a very low interference level may be considered to have a more resistant (or stronger) receiving performance as compared to regions having higher interference levels. Additionally, when using a system inserting and transmitting known data in the data group, wherein the known data are known based upon an agreement between the transmitting system and the receiving system, and when consecutively long known data are to be periodically inserted in the mobile service data, the known data having a predetermined length may be periodically inserted in the region having no interference from the main service data (*i.e.*, a region wherein the main service data are not mixed). However,

due to interference from the main service data, it is difficult to periodically insert known data and also to insert consecutively long known data to a region having interference from the main service data.

**[00116]** Referring to FIG. 5, M/H block 4 (B4) to M/H block 7 (B7) correspond to regions without interference of the main service data. M/H block 4 (B4) to M/H block 7 (B7) within the data group shown in FIG. 5 correspond to a region where no interference from the main service data occurs. In this example, a long known data sequence is inserted at both the beginning and end of each M/H block. In the description of the present invention, the region including M/H block 4 (B4) to M/H block 7 (B7) will be referred to as "region A (=B4+B5+B6+B7)". As described above, when the data group includes region A having a long known data sequence inserted at both the beginning and end of each M/H block, the receiving system is capable of performing equalization by using the channel information that can be obtained from the known data. Therefore, the strongest equalizing performance may be yielded (or obtained) from one of region A to region D.

**[00117]** In the example of the data group shown in FIG. 5, M/H block 3 (B3) and M/H block 8 (B8) correspond to a region having little interference from the main service data. Herein, a long known data sequence is inserted in only one side of each M/H block B3 and B8. More specifically, due to the interference from the main service data, a long known data sequence is inserted at the end of M/H block 3 (B3), and another long known data sequence is inserted at the beginning of M/H block 8 (B8). In the present invention, the region including M/H block 3 (B3) and M/H block 8 (B8) will be referred to as "region B(=B3+B8)". As described above, when the data group includes region B having a long known data sequence inserted at only one side (beginning or end) of each M/H block, the receiving system is capable of performing

equalization by using the channel information that can be obtained from the known data. Therefore, a stronger equalizing performance as compared to region C/D may be yielded (or obtained).

**[00118]** Referring to FIG. 5, M/H block 2 (B2) and M/H block 9 (B9) correspond to a region having more interference from the main service data as compared to region B. A long known data sequence cannot be inserted in any side of M/H block 2 (B2) and M/H block 9 (B9). Herein, the region including M/H block 2 (B2) and M/H block 9 (B9) will be referred to as "region C(=B2+B9)". Finally, in the example shown in FIG. 5, M/H block 1 (B1) and M/H block 10 (B10) correspond to a region having more interference from the main service data as compared to region C. Similarly, a long known data sequence cannot be inserted in any side of M/H block 1 (B1) and M/H block 10 (B10).

**[00119]** Herein, the region including M/H block 1 (B1) and M/H block 10 (B10) will be referred to as "region D (=B1+B10)". Since region C/D is spaced further apart from the known data sequence, when the channel environment undergoes frequent and abrupt changes, the receiving performance of region C/D may be deteriorated.

**[00120]** FIG. 7 illustrates a data structure prior to data interleaving. More specifically, FIG. 7 illustrates an example of 118 data packets being allocated to a data group. FIG. 7 shows an example of a data group consisting of 118 data packets, wherein, based upon a reference packet (e.g., a 1<sup>st</sup> packet (or data segment) or 157<sup>th</sup> packet (or data segment) after a field synchronization signal), when allocating data packets to a VSB frame, 37 packets are included before the reference packet and 81 packets (including the reference packet) are included afterwards.

**[00121]** In other words, with reference to FIG. 5, a field synchronization signal is placed (or assigned) between M/H

block 2 (B2) and M/H block 3 (B3). Accordingly, this indicates that the slot has an off-set of 37 data packets with respect to the corresponding VSB field.

**[00122]** The size of the data groups, number of hierarchical regions within the data group, the size of each region, the number of M/H blocks included in each region, the size of each M/H block, and so on described above are merely exemplary. Therefore, the present invention will not be limited to the examples described above.

**[00123]** FIG. 9 illustrates an exemplary assignment order of data groups being assigned to one of 5 sub-frames, wherein the 5 sub-frames configure an M/H frame. For example, the method of assigning data groups may be identically applied to all M/H frames or differently applied to each M/H frame. Furthermore, the method of assigning data groups may be identically applied to all sub-frames or differently applied to each sub-frame. At this point, when it is assumed that the data groups are assigned using the same method in all sub-frames of the corresponding M/H frame, the total number of data groups being assigned to an M/H frame is equal to a multiple of '5'.

**[00124]** According to the embodiment of the present invention, a plurality of consecutive data groups is assigned to be spaced as far apart from one another as possible within the M/H frame. Thus, the system can be capable of responding promptly and effectively to any burst error that may occur within a sub-frame.

**[00125]** For example, when it is assumed that 3 data groups are assigned to a sub-frame, the data groups are assigned to a 1<sup>st</sup> slot (Slot #0), a 5<sup>th</sup> slot (Slot #4), and a 9<sup>th</sup> slot (Slot #8) in the sub-frame, respectively. FIG. 9 illustrates an example of assigning 16 data groups in one sub-frame using the above-described pattern (or rule). In other words, each data group is serially assigned to 16 slots corresponding to the

following numbers: 0, 8, 4, 12, 1, 9, 5, 13, 2, 10, 6, 14, 3, 11, 7, and 15.

[00126] Equation 1 below shows the above-described rule (or pattern) for assigning data groups in a sub-frame.

[00127] Equation 1

$$j = (4i + 0) \bmod 16$$

Herein,  
 $0 = 0$  if  $i < 4$ ,  
 $0 = 2$  else if  $i < 8$ ,  
 $0 = 1$  else if  $i < 12$ ,  
 $0 = 3$  else.

[00128] Herein,  $j$  indicates the slot number within a sub-frame. The value of  $j$  may range from 0 to 15 (i.e.,  $0 \leq j \leq 15$ ). Also, value of  $i$  indicates the data group number. The value of  $i$  may range from 0 to 15 (i.e.,  $0 \leq i \leq 15$ ).

[00129] In the present invention, a collection of data groups included in an M/H frame will be referred to as a "parade". Based upon the RS frame mode, the parade transmits data of one or two RS frames.

[00130] The mobile service data within one RS frame may be assigned (or mapped) either to all of regions A/B/C/D within the corresponding data group, or to at least one of regions A/B/C/D. In the embodiment of the present invention, the mobile service data within one RS frame may be assigned (or mapped) either to all of regions A/B/C/D, or to at least one of regions A/B and regions C/D. If the mobile service data are assigned to the latter case (i.e., one of regions A/B and regions C/D), the RS frame being assigned to regions A/B and the RS frame being assigned to regions C/D within the corresponding data group are different from one another. In the description of the present invention, the RS frame being assigned to regions A/B within the corresponding data group will be referred to as a "primary RS frame", and the RS frame being assigned to regions C/D within the corresponding data group will be referred to as a "secondary RS frame", for

simplicity. Also, the primary RS frame and the secondary RS frame form (or configure) one parade. More specifically, when the mobile service data within one RS frame are assigned either to all of regions A/B/C/D within the corresponding data group, one parade transmits one RS frame. In this case, also the RS frame will be referred to as a "primary RS frame". Conversely, when the mobile service data within one RS frame are assigned either to at least one of regions A/B and regions C/D, one parade may transmit up to 2 RS frames.

[00131] More specifically, the RS frame mode indicates whether a parade transmits one RS frame, or whether the parade transmits two RS frames.

[00132] Table 1 below shows an example of the RS frame mode.

[00133] Table 1

| RS frame mode (2 bits) | Description                                                                                                                         |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| 00                     | There is only one primary RS frame for all group regions                                                                            |
| 01                     | There are two separate RS frames.<br>- Primary RS frame for group regions A and B<br>- Secondary RS frame for group regions C and D |
| 10                     | Reserved                                                                                                                            |
| 11                     | Reserved                                                                                                                            |

[00134] Table 1 illustrates an example of allocating 2 bits in order to indicate the RS frame mode. For example, referring to Table 1, when the RS frame mode value is equal to '00', this indicates that one parade transmits one RS frame. And, when the RS frame mode value is equal to '01', this indicates that one parade transmits two RS frames, i.e., the primary RS frame and the secondary RS frame. More specifically, when the

RS frame mode value is equal to '01', data of the primary RS frame for regions A/B are assigned and transmitted to regions A/B of the corresponding data group. Similarly, data of the secondary RS frame for regions C/D are assigned and transmitted to regions C/D of the corresponding data group.

**[00135]** As described in the assignment of data groups, the parades are also assigned to be spaced as far apart from one another as possible within the sub-frame. Thus, the system can be capable of responding promptly and effectively to any burst error that may occur within a sub-frame.

**[00136]** Furthermore, the method of assigning parades may be identically applied to all sub-frames or differently applied to each sub-frame. According to the embodiment of the present invention, the parades may be assigned differently for each M/H frame and identically for all sub-frames within an M/H frame. More specifically, the M/H frame structure may vary by M/H frame units. Thus, an ensemble rate may be adjusted on a more frequent and flexible basis.

**[00137]** FIG. 10 illustrates an example of multiple data groups of a single parade being assigned (or allocated) to an M/H frame.

**[00138]** More specifically, FIG. 10 illustrates an example of a plurality of data groups included in a single parade, wherein the number of data groups included in a sub-frame is equal to '3', being allocated to an M/H frame. Referring to FIG. 10, 3 data groups are sequentially assigned to a sub-frame at a cycle period of 4 slots. Accordingly, when this process is equally performed in the 5 sub-frames included in the corresponding M/H frame, 15 data groups are assigned to a single M/H frame. Herein, the 15 data groups correspond to data groups included in a parade. Therefore, since one sub-frame is configured of 4 VSB frame, and since 3 data groups are included in a sub-frame, the data group of the

corresponding parade is not assigned to one of the 4 VSB frames within a sub-frame.

**[00139]** For example, when it is assumed that one parade transmits one RS frame, and that a RS frame encoder located in a later block performs RS-encoding on a payload of the corresponding RS frame, thereby adding 24 bytes of parity data to the corresponding RS frame payload and transmitting the processed RS frame, the parity data occupy approximately 11.37% ( $=24/(187+24) \times 100$ ) of the total code word length. Meanwhile, when one sub-frame includes 3 data groups, and when the data groups included in the parade are assigned, as shown in FIG. 10, a total of 15 data groups form an RS frame. Accordingly, even when an error occurs in an entire data group due to a burst noise within a channel, the percentile is merely 6.67% ( $=1/15 \times 100$ ). Therefore, the receiving system may correct all errors by performing an erasure RS decoding process. More specifically, when the erasure RS decoding is performed, a number of channel errors corresponding to the number of RS parity bytes may be corrected. By doing so, the receiving system may correct the error of at least one data group within one parade. Thus, the minimum burst noise length correctable by a RS frame is over 1 VSB frame.

**[00140]** Meanwhile, when data groups belonging to a parade are assigned as shown in FIG. 10, either main service data may be assigned between each data group, or data groups corresponding to different parades may be assigned between each data group. More specifically, data groups corresponding to multiple parades may be assigned to one M/H frame.

**[00141]** Basically, the method of assigning data groups corresponding to multiple parades is very similar to the method of assigning data groups corresponding to a single parade. In other words, data groups included in other parades that are to be assigned to an M/H frame are also respectively assigned according to a cycle period of 4 slots.

[00142] At this point, data groups of a different parade may be sequentially assigned to the respective slots in a circular method. Herein, the data groups are assigned to slots starting from the ones to which data groups of the previous parade have not yet been assigned.

[00143] For example, when it is assumed that data groups corresponding to a parade are assigned as shown in FIG. 10, data groups corresponding to the next parade may be assigned to a sub-frame starting either from the 12<sup>th</sup> slot of a sub-frame. However, this is merely exemplary. In another example, the data groups of the next parade may also be sequentially assigned to a different slot within a sub-frame at a cycle period of 4 slots starting from the 3<sup>rd</sup> slot.

[00144] As described above, data groups of multiple parades may be assigned to a single M/H frame, and, in each sub-frame, the data groups are serially allocated to a group space having 4 slots from left to right.

[00145] Therefore, a number of groups of one parade per sub-frame (NOG) may correspond to any one integer from '1' to '8'. Herein, since one M/H frame includes 5 sub-frames, the total number of data groups within a parade that can be allocated to an M/H frame may correspond to any one multiple of '5' ranging from '5' to '40'.

[00146] As described above, an M/H frame is divided into 5 sub-frames. Data groups corresponding to a plurality of parades co-exist in each sub-frame. Herein, the data groups corresponding to each parade are grouped by M/H frame units, thereby configuring a single parade.

[00147] Meanwhile, an RS frame according to the present invention includes an RS frame payload, RS parity data added at bottom of each column of the RS frame payload, and CRC data added at left end of each row of the RS frame payload having the RS parity data.

**[00148]** The RS frame payload has the size of  $N$  (row)  $\times$  187 (column), as shown in FIG. 11. Herein,  $N$  represents the length of a row (i.e., number of columns), and 187 corresponds to the length of a column (i.e., number of rows).

**[00149]** In the present invention, for convenience of description, each row of the  $N$  bytes will be referred to as M/H service data packet (or M/H TP packet).

**[00150]** Each M/H service data packet within the RS frame payload includes M/H header (or MH TP header) of 2 bytes, a stuffing region of  $k$  bytes, and M/H payload of  $N-2-k$  bytes as shown in FIG. 12. At this time,  $k$  has a value of 0 or a value greater than 0. The M/H payload includes IP datagrams of signaling table information and/or mobile service data according to an embodiment of the present invention. In this case, the M/H header of 2 bytes is only one example, and corresponding bytes can be varied depending on a designer. Accordingly, the present invention will not be limited to such example.

**[00151]** At this time, as the M/H service data packet includes M/H header, the M/H header may not reach  $N$  bytes.

**[00152]** In this case, stuffing bytes can be assigned to the remaining payload part of the corresponding M/H service data packet. For example, after program table information is assigned to one M/H service data packet, if the length of the M/H service data packet is  $N-20$  bytes including the M/H header, the stuffing bytes can be assigned to the remaining 20 bytes. In this case, the value  $k$  becomes 20, and the M/H payload region within the corresponding M/H service data packet includes  $N-2-20$  bytes.

**[00153]** The RS frame payload is generated by collecting signaling table information corresponding to one or more mobile services and/or IP datagram of the mobile service data. For example, signaling table information for two kinds of mobile services called news (for example, IP datagram for

mobile service 1) and the stocks (for example, IP datagram for mobile service 2) and IP datagram of mobile service data can be included in one RS frame payload.

**[00154]** More specifically, in the transmitting system (e.g., mobile broadcast station), the mobile service data (e.g., A/V steaming) are packetized based upon a real time protocol (RTP) method. The RTP packet is then packetized once again based upon a user datagram protocol (UDP) method. Thereafter, the RTP/UDP packet is in turn packetized based upon an IP method, thereby being packetized into RTP/UDP/IP packet data. In the description of the present invention, the packetized RTP/UDP/IP packet data will be referred to as an IP datagram for simplicity.

**[00155]** Furthermore, service information for receiving mobile services may be provided in the form of a signaling table. And, a service signaling channel transmitting such signaling table is packetized based upon a UDP method. And, the packetized UDP data are then packetized based upon an IP method, thereby being packetized into UDP/IP data. In the description of the present invention, the packetized UDP/IP packet data will also be referred to as an IP datagram for simplicity. According to an embodiment of the present invention, the service signaling channel is encapsulated into an IP datagram having a well-known destination IP address and a well-known destination UDP port number.

**[00156]** More specifically, one RS frame payload includes an IP datagram of mobile service data for at least one or more mobile services. Furthermore, the RS frame payload includes an IP datagram of a service signaling channel for receiving the mobile service data.

**[00157]** According to the embodiment of the present invention, among a service map table (SMT), a guide access table (GAT), a cell information table (CIT), a service labeling table (SLT), and a rating region table (RRT), the present invention

transmits at least one signaling table through the service signaling channel. Herein, the signaling tables presented in the embodiment of the present invention are merely examples for facilitating the understanding of the present invention. Therefore, the present invention is not limited only to the exemplary signaling tables that can be transmitted through the service signaling channel.

**[00158]** The SMT provides signaling information on ensemble levels. Also, each SMT provides IP access information for each mobile service belonging to the corresponding ensemble including each SMT. Furthermore, the SMT provides IP stream component level information required for the corresponding mobile service. The RRT transmits information on region and consultation organs for program ratings. More specifically, the RRT provides content advisory rating information. The GAT provides information on SG providers, which transmit the service guides. Also, the GAT provides service guide bootstrapping information required for accessing the SG. The CIT provides channel information of each cell, which corresponds to the frequency domain of a broadcast signal. Herein, a cell refers to a scope affected (or influenced) by a transmitter based upon a physical frequency in a multi-frequency network (MFN) environment (or condition). More specifically, the CIT provides information on a carrier wave frequency of an adjacent cell in the current transmitter (or transmitting system). Therefore, based upon the CIT information, a receiver (or receiving system) can travel from one transmitter's (or exciter's) coverage area to another. The SLT provides minimum required information for an exclusive usage of a channel scan process. More specifically, according to the embodiment of the present invention, other than the SMT, by using the SLT for the exclusive usage of the channel scan process, so as to configure a set of minimum information for

the channel scan process, the channel scanning speed may be increased.

**[00159]** According to an embodiment of the present invention, each signaling table is divided into at least one section. Then, each section is encapsulated to a UDP/IP header, thereby being transmitted through the service signaling channel. In this case, the number of UDP/IP packets being transmitted through the service signaling channel may vary based upon the number of signaling tables being transmitted through the service signaling channel and the number of sections in each signaling table.

**[00160]** At this point, all UDP/IP packets transmitted through the service signaling channel have the same number of well-known target IP addresses and well-known target UDP port numbers. For example, when it is assumed that the SMT, RRT, and GAT are transmitted through the service signaling channel, the target IP address and target UDP port number of all UDP/IP packets transmitting the SMT, RRT, and GAT are identical to one another. Furthermore, the target IP address and the target UDP port number respectively correspond to well-known values, i.e., values pre-known by the receiving system based upon an agreement between the receiving system and the transmitting system.

**[00161]** Therefore, the identification of each signaling table included in the service signaling data is performed by a table identifier. The table identifier may correspond to a table\_id field existing in the corresponding signaling table or in the header of the corresponding signaling table section. And, when required, identification may be performed by further referring to a table\_id\_extension field.

**[00162]** FIG. 12 is a diagram illustrating examples of fields allocated to the M/H header region within the M/H service data packet according to the present invention. Examples of the

fields include type\_indicator field, error\_indicator field, stuff\_indicator field, and pointer field.

**[00163]** The type\_indicator field can allocate 3 bits, for example, and represents a type of data allocated to payload within the corresponding M/H service data packet. In other words, the type\_indicator field indicates whether data of the payload is IP datagram or program table information. At this time, each data type constitutes one logical channel. In the logical channel which transmits the IP datagram, several mobile services are multiplexed and then transmitted. Each mobile service undergoes demultiplexing in the IP layer.

**[00164]** The error\_indicator field can allocate 1 bit, for example, and represents whether the corresponding M/H service data packet has an error. For example, if the error\_indicator field has a value of 0, it means that there is no error in the corresponding M/H service data packet. If the error\_indicator field has a value of 1, it means that there may be an error in the corresponding M/H service data packet. According to an embodiment of the present invention, a value of zero are indicated and transmitted on all error\_indicator fields within the RS frame.

**[00165]** The stuff\_indicator field can allocate 1 bit, for example, and represents whether stuffing byte exists in payload of the corresponding M/H service data packet. For example, if the stuff\_indicator field has a value of 0, it means that there is no stuffing byte in the corresponding M/H service data packet. If the stuff\_indicator field has a value of 1, it means that stuffing byte exists in the corresponding M/H service data packet.

**[00166]** The pointer field can allocate 11 bits, for example, and represents position information where new data (i.e., new signaling information or new IP datagram) starts in the corresponding M/H service data packet.

**[00167]** For example, if IP datagram for mobile service 1 and IP datagram for mobile service 2 are allocated to the first M/H service data packet within the RS frame payload as illustrated in FIG. 11, the pointer field value represents the start position of the IP datagram for mobile service 2 within the M/H service data packet.

**[00168]** Also, if there is no new data in the corresponding M/H service data packet, the corresponding field value is expressed as a maximum value exemplarily. According to the embodiment of the present invention, since 11 bits are allocated to the pointer field, if 2047 is expressed as the pointer field value, it means that there is no new data in the packet. The point where the pointer field value is 0 can be varied depending on the type\_indicator field value and the stuff\_indicator field value.

**[00169]** It is to be understood that the order, the position, and the meaning of the fields allocated to the header within the M/H service data packet illustrated in FIG. 12 are exemplarily illustrated for understanding of the present invention. Since the order, the position and the meaning of the fields allocated to the header within the M/H service data packet and the number of additionally allocated fields can easily be modified by those skilled in the art, the present invention will not be limited to the above example.

**[00170]** FIG. 13(a) and FIG. 13(b) illustrate another examples of RS frame payload according to the present invention. FIG. 13(a) illustrates an example of primary RS frame payload to be allocated to regions A/B within the data group, and FIG. 13(b) illustrates an example of secondary RS frame payload to be allocated to regions C/D within the data group.

**[00171]** In FIG. 13(a) and FIG. 13(b), a column length (i.e., the number of rows) of the RS frame payload to be allocated to the regions A/B and a column length (i.e., the number of rows)

of the RS frame payload to be allocated to the regions C/D are 187 equally. However, row lengths (i.e, the number of columns) may be different from each other.

**[00172]** According to the embodiment of the present invention, when the row length of the primary RS frame payload to be allocated to the regions A/B within the data group is N1 bytes and the row length of the secondary RS frame payload to be allocated to the regions C/D within the data group is N2 bytes, a condition of N1>N2 is satisfied.

**[00173]** In this case, N1 and N2 can be varied depending on the transmission parameter or a region of the data group, to which the corresponding RS frame payload will be transmitted.

**[00174]** For convenience of the description, each row of the N1 and N2 bytes will be referred to as the M/H service data packet. In the present invention, the primary RS frame payload for the regions A/B within the data group and the secondary RS frame payload for the regions C/D within the data group can include at least one of IP datagrams of signaling table information and mobile service data. Also, one RS frame payload can include IP datagram corresponding to one or more mobile services.

**[00175]** Corresponding parts of FIG. 11 can be applied to the other parts, which are not described in FIG. 13(a) and FIG. 13(b).

**[00176]** Meanwhile, the value of N, which corresponds to the number of columns within an RS frame payload, can be decided according to Equation 2.

**[00177]** Equation 2

$$N = \left\lfloor \frac{5 \times NoG \times PL}{187 + P} \right\rfloor - 2$$

**[00178]** Herein, NoG indicates the number of data groups assigned to a sub-frame. PL represents the number of SCCC payload data bytes assigned to a data group. And, P signifies

the number of RS parity data bytes added to each column of the RS frame payload. Finally,  $\lfloor X \rfloor$  is the greatest integer that is equal to or smaller than  $X$ .

**[00179]** More specifically, in Equation 2, PL corresponds to the length of an RS frame portion. The value of PL is equivalent to the number of SCCC payload data bytes that are assigned to the corresponding data group. Herein, the value of PL may vary depending upon the RS frame mode, SCCC block mode, and SCCC outer code mode. Table 2 to Table 5 below respectively show examples of PL values, which vary in accordance with the RS frame mode, SCCC block mode, and SCCC outer code mode. The SCCC block mode and the SCCC outer code mode will be described in detail in a later process.

**[00180]** Table 2

| SCCC outer code mode |                 |                 |                 | PL   |
|----------------------|-----------------|-----------------|-----------------|------|
| for<br>Region A      | for<br>Region B | for<br>Region C | for<br>Region D |      |
| 00                   | 00              | 00              | 00              | 9624 |
| 00                   | 00              | 00              | 01              | 9372 |
| 00                   | 00              | 01              | 00              | 8886 |
| 00                   | 00              | 01              | 01              | 8634 |
| 00                   | 01              | 00              | 00              | 8403 |
| 00                   | 01              | 00              | 01              | 8151 |
| 00                   | 01              | 01              | 00              | 7665 |
| 00                   | 01              | 01              | 01              | 7413 |
| 01                   | 00              | 00              | 00              | 7023 |
| 01                   | 00              | 00              | 01              | 6771 |
| 01                   | 00              | 01              | 00              | 6285 |
| 01                   | 00              | 01              | 01              | 6033 |
| 01                   | 01              | 00              | 00              | 5802 |

|        |    |    |    |          |
|--------|----|----|----|----------|
| 01     | 01 | 00 | 01 | 5550     |
| 01     | 01 | 01 | 00 | 5064     |
| 01     | 01 | 01 | 01 | 4812     |
| Others |    |    |    | Reserved |

**[00181]** Table 2 shows an example of the PL values for each data group within an RS frame, wherein each PL value varies depending upon the SCCC outer code mode, when the RS frame mode value is equal to '00', and when the SCCC block mode value is equal to '00'.

**[00182]** For example, when it is assumed that each SCCC outer code mode value of regions A/B/C/D within the data group is equal to '00' (i.e., the block processor 302 of a later block performs encoding at a coding rate of 1/2), the PL value within each data group of the corresponding RS frame may be equal to 9624 bytes. More specifically, 9624 bytes of mobile service data within one RS frame may be assigned to regions A/B/C/D of the corresponding data group.

**[00183]** Table 3

| SCCC outer code mode | PL       |
|----------------------|----------|
| 00                   | 9624     |
| 01                   | 4812     |
| Others               | Reserved |

**[00184]** Table 3 shows an example of the PL values for each data group within an RS frame, wherein each PL value varies depending upon the SCCC outer code mode, when the RS frame mode value is equal to '00', and when the SCCC block mode value is equal to '01'.

**[00185]** Table 4

| SCCC outer code mode |              | PL |
|----------------------|--------------|----|
| For Region A         | for Region B |    |

|        |    |          |
|--------|----|----------|
| 00     | 00 | 7644     |
| 00     | 01 | 6423     |
| 01     | 00 | 5043     |
| 01     | 01 | 3822     |
| Others |    | Reserved |

**[00186]** Table 4 shows an example of the PL values for each data group within a primary RS frame, wherein each PL value varies depending upon the SCCC outer code mode, when the RS frame mode value is equal to '01', and when the SCCC block mode value is equal to '00'. For example, when each SCCC outer code mode value of regions A/B is equal to '00', 7644 bytes of mobile service data within a primary RS frame may be assigned to regions A/B of the corresponding data group.

**[00187]** Table 5

| SCCC outer code mode |              | PL       |
|----------------------|--------------|----------|
| For Region C         | for Region D |          |
| 00                   | 00           | 1980     |
| 00                   | 01           | 1728     |
| 01                   | 00           | 1242     |
| 01                   | 01           | 990      |
| Others               |              | Reserved |

**[00188]** Table 5 shows an example of the PL values for each data group within a secondary RS frame, wherein each PL value varies depending upon the SCCC outer code mode, when the RS frame mode value is equal to '01', and when the SCCC block mode value is equal to '00'. For example, when each SCCC outer code mode value of regions C/D is equal to '00', 1980 bytes of mobile service data within a secondary RS frame may be assigned to regions C/D of the corresponding data group.

[00189] Meanwhile, a predetermined portion of each data group (i.e., 37 bytes/data group) is used for delivering (or sending) FIC information (i.e., one FIC segment), wherein the FIC information is separately encoded from the RS-encoding process on the mobile service data.

[00190] Also, the concept of an M/H ensemble is applied in the embodiment of the present invention, thereby defining a collection (or group) of services. Each M/H ensemble carries the same QoS and is coded with the same FEC code. Herein, the M/H ensemble is used as the same meaning of ensemble. Furthermore, each ensemble has a unique identifier (i.e., ensemble ID) and corresponds to consecutive RS frames.

[00191] A transmitting/receiving system according to an embodiment of the present invention operates (or manages) two data channels. One data channel is an RS frame data channel for transmitting contents and the other data channel is an FIC (Fast Information Channel) for service acquisition. The present invention is intended that mapping (binding) information between an ensemble and a mobile service is signaled using an FIC chunk and the FIC chunk is transmitted through the FIC by being segmented into an FIC segment unit, whereby the receiving system can perform fast service acquisition.

[00192]

[00193] Transmitter

[00194] FIG. 14 illustrates a block diagram showing an example of the transmitter according to an embodiment of the present invention. Herein, the transmitter includes a packet jitter mitigator 220, a pre-processor 230, a packet multiplexer 240, a post-processor 250, a synchronization (sync) multiplexer 260, and a transmission unit 270.

[00195] At this point, main service data are inputted to the packet jitter mitigator 220 and mobile service data are inputted to the pre-processor 230 according to an embodiment

of the present invention. Furthermore, main service data packets including main service data can be inputted to the packet jitter mitigator 220 and mobile service data packets including mobile service data can be inputted to the pre-processor 230 according to another embodiment of the present invention.

**[00196]** The pre-processor 230 performs an additional encoding process of the mobile service data inputted. The pre-processor 230 also performs a process of configuring a data group so that the data group may be positioned at a specific place in accordance with the purpose of the data, which are to be transmitted on a transmission frame. This is to enable the mobile service data to respond swiftly and strongly against noise and channel changes. FIG. 15 illustrates a block diagram showing the structure of a pre-processor 230 according to the present invention. Herein, the pre-processor 230 includes an M/H frame encoder 301, a block processor 302, a group formatter 303, a signaling encoder 304, and a packet formatter 305.

**[00197]** The M/H frame encoder 301, which is included in the pre-processor 230 having the above-described structure, data-randomizes the mobile service data inputted, thereby forming at least one RS frame belonging to an ensemble. The M/H frame encoder 301 may include at least one RS frame encoder. More specifically, RS frame encoders may be provided in parallel, wherein the number of RS frame encoders is equal to the number of parades within the M/H frame. As described above, the M/H frame is a basic time cycle period for transmitting at least one parade. Also, each parade consists of one or two RS frames.

**[00198]** FIG. 16 illustrates a conceptual block diagram of the M/H frame encoder 301 according to an embodiment of the present invention. The M/H frame encoder 301 includes an input demultiplexer (DEMUX) 309, M number of RS frame encoders 310

to 31M-1, and an output multiplexer (MUX) 320. Herein, M represent the number of parades included in one M/H frame.

**[00199]** The demultiplexer 309 output the inputted mobile service data packet to a corresponding RS frame encoder among M number of RS frame encoders in ensemble units.

**[00200]** According to an embodiment of the present invention, each RS frame encoder forms an RS frame payload using mobile service data inputted and performs at least one of an error correction encoding process and an error detection encoding process in RS frame payload units, thereby forming an RS frame. Also, each RS frame encoder divides the RS frame into a plurality of portions, in order to assign the error-correction-encoded RS frame data to a plurality of data groups. Based upon the RS frame mode of Table 1, data within one RS frame may be assigned either to all of regions A/B/C/D within multiple data groups, or to at least one of regions A/B and regions C/D within multiple data groups.

**[00201]** When the RS frame mode value is equal to '01', i.e., when the data of the primary RS frame are assigned to regions A/B of the corresponding data group and data of the secondary RS frame are assigned to regions C/D of the corresponding data group, each RS frame encoder creates a primary RS frame and a secondary RS frame for each parade. Conversely, when the RS frame mode value is equal to '00', when the data of the primary RS frame are assigned to all of regions A/B/C/D, each RS frame encoder creates a RS frame (i.e., a primary RS frame) for each parade.

**[00202]** Also, each RS frame encoder divides each RS frame into several portions. Each portion of the RS frame is equivalent to a data amount that can be transmitted by a data group. The output multiplexer (MUX) 320 multiplexes portions within M number of RS frame encoders 310 to 310M-1 are multiplexed and then outputted to the block processor 302.

**[00203]** For example, if one parade transmits two RS frames, portions of primary RS frames within M number of RS frame encoders 310 to 310M-1 are multiplexed and outputted. Thereafter, portions of secondary RS frames within M number of RS frame encoders 310 to 310M-1 are multiplexed and transmitted.

**[00204]** FIG. 17 illustrates a detailed block diagram of an RS frame encoder among a plurality of RS frame encoders within an M/H frame encoder.

**[00205]** One RS frame encoder may include a primary encoder 410 and a secondary encoder 420. Herein, the secondary encoder 420 may or may not operate based upon the RS frame mode. For example, when the RS frame mode value is equal to '00', as shown in Table 1, the secondary encoder 420 does not operate.

**[00206]** The primary encoder 410 may include a data randomizer 411, a Reed-Solomon-cyclic redundancy check (RS-CRC) encoder (412), and a RS frame divider 413. And, the secondary encoder 420 may also include a data randomizer 421, a RS-CRC encoder (422), and a RS frame divider 423.

**[00207]** More specifically, the data randomizer 411 of the primary encoder 410 receives mobile service data belonging to a primary ensemble outputted from the output demultiplexer (DEMUX) 309. Then, after randomizing the received mobile service data, the data randomizer 411 outputs the randomized data to the RS-CRC encoder 412.

**[00208]** The RS-CRC encoder 412 performs FEC (Forward Error Correction) encoding on the randomized mobile service data using at least one of a Reed-Solomon (RS) code and a cyclic redundancy check (CRC) code and outputs to the RS frame divider 413.

**[00209]** The RS-CRC encoder 412 groups a plurality of mobile service data that is randomized and inputted, so as to form a RS frame payload. Then, the RS-CRC encoder 412 performs at least one of an error correction encoding process and an error

detection encoding process in RS frame payload units, thereby forming an RS frame. Accordingly, robustness may be provided to the mobile service data, thereby scattering group error that may occur during changes in a frequency environment, thereby enabling the mobile service data to respond to the frequency environment, which is extremely vulnerable and liable to frequent changes.

**[00210]** Also, the RS-CRC encoder 412 groups a plurality of RS frame so as to create a super frame, thereby performing a row permutation process in super frame units. The row permutation process may also be referred to as a "row interleaving process". Hereinafter, the process will be referred to as "row permutation" for simplicity. In the present invention, the row permutation process is optional.

**[00211]** More specifically, when the RS-CRC encoder 412 performs the process of permuting each row of the super frame in accordance with a pre-determined rule, the position of the rows within the super frame before and after the row permutation process is changed. If the row permutation process is performed by super frame units, and even though the section having a plurality of errors occurring therein becomes very long, and even though the number of errors included in the RS frame, which is to be decoded, exceeds the extent of being able to be corrected, the errors become dispersed within the entire super frame. Thus, the decoding ability is even more enhanced as compared to a single RS frame.

**[00212]** At this point, as an example of the present invention, RS-encoding is applied for the error correction encoding process, and a cyclic redundancy check (CRC) encoding is applied for the error detection process in the RS-CRC encoder 412. When performing the RS-encoding, parity data that are used for the error correction are generated. And, when performing the CRC encoding, CRC data that are used for the error detection are generated.

**[00213]** The CRC data generated by CRC encoding may be used for indicating whether or not the mobile service data have been damaged by the errors while being transmitted through the channel. In the present invention, a variety of error detection coding methods other than the CRC encoding method may be used, or the error correction coding method may be used to enhance the overall error correction ability of the receiving system.

**[00214]** FIG. 18(a) and FIG. 18(b) illustrate a process of one or two RS frame being divided into several portions, based upon an RS frame mode value, and a process of each portion being assigned to a corresponding region within the respective data group. According to an embodiment of the present invention, the data assignment within the data group is performed by the group formatter 303.

**[00215]** More specifically, FIG. 18(a) shows an example of the RS frame mode value being equal to '00'. Herein, only the primary encoder 410 of FIG. 17 operates, thereby forming one RS frame for one parade. Then, the RS frame is divided into several portions, and the data of each portion are assigned to regions A/B/C/D within the respective data group.

**[00216]** FIG. 18(b) shows an example of the RS frame mode value being equal to '01'. Herein, both the primary encoder 410 and the secondary encoder 420 of FIG. 17 operate, thereby forming two RS frames for one parade, i.e., one primary RS frame and one secondary RS frame. Then, the primary RS frame is divided into several portions, and the secondary RS frame is divided into several portions. At this point, the data of each portion of the primary RS frame are assigned to regions A/B within the respective data group. And, the data of each portion of the secondary RS frame are assigned to regions C/D within the respective data group.

**[00217]**

**[00218]** Detailed Description of the RS Frame

**[00219]** FIG. 19(a) illustrates an example of an RS frame being generated from the RS-CRC encoder 412 according to the present invention.

**[00220]** When the RS frame payload is formed, as shown in FIG. 19(a), the RS-CRC encoder 412 performs a  $(N_c, K_c)$ -RS encoding process on each column, so as to generate  $N_c - K_c (=P)$  number of parity bytes. Then, the RS-CRC encoder 412 adds the newly generated  $P$  number of parity bytes after the very last byte of the corresponding column, thereby creating a column of  $(187+P)$  bytes.

**[00221]** Herein, as shown in FIG. 19(a),  $K_c$  is equal to 187 (*i.e.*,  $K_c=187$ ), and  $N_c$  is equal to  $187+P$  (*i.e.*,  $N_c=187+P$ ).

**[00222]** Herein, the value of  $P$  may vary depending upon the RS code mode. Table 6 below shows an example of an RS code mode, as one of the RS encoding information.

**[00223]** Table 6

| RS code mode | RS code   | Number of Parity Bytes (P) |
|--------------|-----------|----------------------------|
| 00           | (211,187) | 24                         |
| 01           | (223,187) | 36                         |
| 10           | (235,187) | 48                         |
| 11           | Reserved  | Reserved                   |

**[00224]** Table 6 shows an example of 2 bits being assigned in order to indicate the RS code mode. The RS code mode represents the number of parity bytes corresponding to the RS frame payload.

**[00225]** For example, when the RS code mode value is equal to '10',  $(235,187)$ -RS-encoding is performed on the RS frame payload of FIG. 19(a), so as to generate 48 parity data bytes. Thereafter, the 48 parity bytes are added after the last data byte of the corresponding column, thereby creating a column of 235 data bytes.

**[00226]** When the RS frame mode value is equal to '00' in Table 1 (i.e., when the RS frame mode indicates a single RS frame), only the RS code mode of the corresponding RS frame is indicated. However, when the RS frame mode value is equal to '01' in Table 1 (i.e., when the RS frame mode indicates multiple RS frames), the RS code mode corresponding to a primary RS frame and a secondary RS frame. More specifically, it is preferable that the RS code mode is independently applied to the primary RS frame and the secondary RS frame.

**[00227]** When such RS encoding process is performed on all N number of columns, a size of  $N(\text{row}) \times (187+P)(\text{column})$  bytes may be generated, as shown in FIG. 19(b).

**[00228]** Each row of the RS frame payload is configured of N bytes. However, depending upon channel conditions between the transmitting system and the receiving system, error may be included in the RS frame payload. When errors occur as described above, CRC data (or CRC code or CRC checksum) may be used on each row unit in order to verify whether error exists in each row unit.

**[00229]** The RS-CRC encoder 412 may perform CRC encoding on the mobile service data being RS encoded so as to create (or generate) the CRC data. The CRC data being generated by CRC encoding may be used to indicate whether the mobile service data have been damaged while being transmitted through the channel.

**[00230]** The present invention may also use different error detection encoding methods other than the CRC encoding method. Alternatively, the present invention may use the error correction encoding method to enhance the overall error correction ability of the receiving system.

**[00231]** FIG. 19(c) illustrates an example of using a 2-byte (i.e., 16-bit) CRC checksum as the CRC data. Herein, a 2-byte CRC checksum is generated for N number of bytes of each row, thereby adding the 2-byte CRC checksum at the end of the N

number of bytes. Thus, each row is expanded to (N+2) number of bytes.

[00232] Equation 3 below corresponds to an exemplary equation for generating a 2-byte CRC checksum for each row being configured of N number of bytes.

[00233] Equation 3

$$g(x) = x^{16} + x^{12} + x^5 + 1$$

[00234] The process of adding a 2-byte checksum in each row is only exemplary. Therefore, the present invention is not limited only to the example proposed in the description set forth herein.

[00235] As described above, when the process of RS encoding and CRC encoding are completed, the (Nx187)-byte RS frame payload is converted into a (N+2)x(187+P)-byte RS frame.

[00236] Based upon an error correction scenario of a RS frame formed as described above, the data bytes within the RS frame are transmitted through a channel in a row direction. At this point, when a large number of errors occur during a limited period of transmission time, errors also occur in a row direction within the RS frame being processed with a decoding process in the receiving system. However, in the perspective of RS encoding performed in a column direction, the errors are shown as being scattered. Therefore, error correction may be performed more effectively. At this point, a method of increasing the number of parity data bytes (P) may be used in order to perform a more intense error correction process. However, using this method may lead to a decrease in transmission efficiency. Therefore, a mutually advantageous method is required. Furthermore, when performing the decoding process, an erasure decoding process may be used to enhance the error correction performance.

[00237] The above description of the present invention corresponds to the processes of forming (or creating) and

encoding an RS frame, when a data group is divided into regions A/B/C/D, and when data of an RS frame are assigned to all of regions A/B/C/D within a plurality of data groups. More specifically, the above description corresponds to an embodiment of the present invention, wherein one RS frame is transmitted using one parade. In this embodiment, the secondary encoder 420 does not operate (or is not active).

**[00238]** Meanwhile, 2 RS frames are transmitting using one parade, the data of the primary RS frame may be assigned to regions A/B within the data groups and be transmitted, and the data of the secondary RS frame may be assigned to regions C/D within the data groups and be transmitted. At this point, the primary encoder 410 receives the mobile service data that are to be assigned to regions A/B within the data groups, forms the primary RS frame payload, and then performs RS-encoding and CRC-encoding on the primary RS frame payload, thereby forming the primary RS frame. Similarly, the secondary encoder 420 receives the mobile service data that are to be assigned to regions C/D within the data groups, forms the secondary RS frame payload, and then performs RS-encoding and CRC-encoding on the secondary RS frame payload thereby forming the secondary RS frame. More specifically, the primary RS frame and the secondary RS frame are generated independently.

**[00239]** FIG. 20 illustrates examples of receiving the mobile service data that are to be assigned to regions A/B within the data group, so as to form the primary RS frame payload, and receives the mobile service data that are to be assigned to regions C/D within the data group, so as to form the secondary RS frame payload, thereby performing error correction encoding and error detection encoding on each of the first and secondary RS frame payloads.

**[00240]** More specifically, FIG. 20(a) illustrates an example of the RS-CRC encoder 412 of the primary encoder 410 receiving mobile service data of the primary ensemble that are to be

assigned to regions A/B within the corresponding data group, so as to create an RS frame payload having the size of  $N1(\text{row}) \times 187(\text{column})$ . Then, in this example, the primary encoder 410 performs RS-encoding on each column of the RS frame payload created as described above, thereby adding  $P1$  number of parity data bytes in each column. Finally, the primary encoder 410 performs CRC-encoding on each row, thereby adding a 2-byte checksum in each row, thereby forming an primary RS frame.

**[00241]** FIG. 20(b) illustrates an example of the RS-CRC encoder 422 of the secondary encoder 420 receiving mobile service data of the secondary ensemble that are to be assigned to regions C/D within the corresponding data group, so as to create an RS frame payload having the size of  $N2(\text{row}) \times 187(\text{column})$ . Then, in this example, the secondary encoder 420 performs RS-encoding on each column of the RS frame payload created as described above, thereby adding  $P2$  number of parity data bytes in each column. Finally, the secondary encoder 420 performs CRC-encoding on each row, thereby adding a 2-byte checksum in each row, thereby forming an secondary RS frame.

**[00242]** At this point, each of the RS-CRC encoders 412 and 422 performs RS frame configuration, error correction encoding, error detection encoding by referring to a transmission parameter, for example, at least one of M/H frame information, FIC information, RS frame information (including RS frame mode information), RS encoding information (including RS code mode information), SCCC information (including SCCC block mode information and SCCC outer code mode information). Furthermore, the transmission parameters should also be transmitted to the receiving system so that the receiving system can perform a normal decoding process. At this point, as an example of the present invention, the transmission parameter is transmitted

through transmission parameter channel (TPC) to a receiving system. The TPC will be described in detail in a later.

**[00243]** The data of the primary RS frame, which is encoded by RS frame units by the RS-CRC encoder 412 of the primary encoder 410, are outputted to the RS frame divider 413. If the secondary encoder 420 also operates in the embodiment of the present invention, the data of the secondary RS frame, which is encoded by RS frame units by the RS-CRC encoder 422 of the secondary encoder 420, are outputted to the RS frame divider 423.

**[00244]** The RS frame divider 413 of the primary encoder 410 divides the primary RS frame into several portions, which are then outputted to the output multiplexer (MUX) 320. Each portion of the primary RS frame is equivalent to a data amount that can be transmitted by one data group. Similarly, the RS frame divider 423 of the secondary encoder 420 divides the secondary RS frame into several portions, which are then outputted to the output multiplexer (MUX) 320.

**[00245]** Hereinafter, the RS frame divider 413 of the primary RS encoder 410 will now be described in detail. Also, in order to simplify the description of the present invention, it is assumed that an RS frame payload having the size of  $N(\text{row}) \times 187(\text{column})$ , as shown in FIG. 19(a) to FIG. 19(c), that  $P$  number of parity data bytes are added to each column by RS-encoding the RS frame payload, and that a 2-byte checksum is added to each row by CRC-encoding the RS frame payload. As a result, an RS frame having the size of  $(N+2) (\text{row}) \times 187+P (\text{column})$  is formed.

**[00246]** Accordingly, the RS frame divider 413 divides (or partitions) the RS frame having the size of  $(N+2) (\text{row}) \times 187+P (\text{column})$  into several portions, each having the size of  $PL$  (wherein  $PL$  corresponds to the length of the RS frame portion).

**[00247]** At this point, as shown in Table 2 to Table 5, the value of PL may vary depending upon the RS frame mode, SCCC block mode, and SCCC outer coder mode. Also, the total number of data bytes of the RS-encoded and CRC-encoded RS frame is equal to or smaller than  $5 \times NoG \times PL$ . In this case, the RS frame is divided (or partitioned) into  $((5 \times NoG) - 1)$  number of portions each having the size of PL and one portion having a size equal to smaller than PL. More specifically, with the exception of the last portion of the RS frame, each of the remaining portions of the RS frame has an equal size of PL.

**[00248]** If the size of the last portion is smaller than PL, a stuffing byte (or dummy byte) may be inserted in order to fill (or replace) the lacking number of data bytes, thereby enabling the last portion of the RS frame to also be equal to PL.

**[00249]** Each portion of an RS frame corresponds to the amount of data that are to be SCCC-encoded and mapped into a single data group of a parade.

**[00250]** FIG. 21(a) and FIG. 21(b) respectively illustrate examples of adding S number of stuffing bytes, when an RS frame having the size of  $(N+2)(row) \times (187+P)(column)$  is divided into  $5 \times NoG$  number of portions, each having the size of PL.

**[00251]** More specifically, the RS-encoded and CRC-encoded RS frame, shown in FIG. 21(a), is divided into several portions, as shown in FIG. 21(b). The number of divided portions at the RS frame is equal to  $(5 \times NoG)$ . Particularly, the first  $((5 \times NoG) - 1)$  number of portions each has the size of PL, and the last portion of the RS frame may be equal to or smaller than PL. If the size of the last portion is smaller than PL, a stuffing byte (or dummy byte) may be inserted in order to fill (or replace) the lacking number of data bytes, as shown in Equation 4 below, thereby enabling the last portion of the RS frame to also be equal to PL.

**[00252]** Equation 4

$$S = (5 \times N_o G \times PL) - ((N + 2) \times (187 + P))$$

[00253] Herein, each portion including data having the size of PL passes through the output multiplexer 320 of the M/H frame encoder 301, which is then outputted to the block processor 302.

[00254] At this point, the mapping order of the RS frame portions to a parade of data groups is not identical with the group assignment order defined in Equation 1. When given the group positions of a parade in an M/H frame, the SCCC-encoded RS frame portions will be mapped in a time order (i.e., in a left-to-right direction).

[00255]

[00256] Block Processor

[00257] Meanwhile, the block processor 302 performs an SCCC outer encoding process on the output of the M/H frame encoder 301. More specifically, the block processor 302 receives the data of each error correction encoded portion. Then, the block processor 302 encodes the data once again at a coding rate of  $1/H$  (wherein  $H$  is an integer equal to or greater than 2 (i.e.,  $H \geq 2$ )), thereby outputting the  $1/H$ -rate encoded data to the group formatter 303. According to the embodiment of the present invention, the input data are encoded either at a coding rate of  $1/2$  (also referred to as "1/2-rate encoding") or at a coding rate of  $1/4$  (also referred to as "1/4-rate encoding"). The data of each portion outputted from the M/H frame encoder 301 may include at least one of mobile service data, RS parity data, CRC data, and stuffing data. However, in a broader meaning, the data included in each portion may correspond to data for mobile services. Therefore, the data included in each portion will all be considered as mobile service data and described accordingly.

[00258] The group formatter 303 inserts the mobile service data SCCC-outer-encoded and outputted from the block processor

302 in the corresponding region within the data group, which is formed in accordance with a pre-defined rule. Also, in association with the data deinterleaving process, the group formatter 303 inserts various place holders (or known data place holders) in the corresponding region within the data group. Thereafter, the group formatter 303 deinterleaves the data within the data group and the place holders.

**[00259]** According to the present invention, with reference to data after being data-interleaved, as shown in FIG. 5, a data groups is configured of 10 M/H blocks (B1 to B10) and divided into 4 regions (A, B, C, and D).

**[00260]** Also, as shown in FIG. 5, when it is assumed that the data group is divided into a plurality of hierarchical regions, as described above, the block processor 302 may encode the mobile service data, which are to be inserted to each region based upon the characteristic of each hierarchical region, at different coding rates.

**[00261]** For example, the block processor 302 may encode the mobile service data, which are to be inserted in region A/B within the corresponding data group, at a coding rate of  $1/2$ . Then, the group formatter 303 may insert the  $1/2$ -rate encoded mobile service data to region A/B. Also, the block processor 302 may encode the mobile service data, which are to be inserted in region C/D within the corresponding data group, at a coding rate of  $1/4$  having higher (or stronger) error correction ability than the  $1/2$ -coding rate. Thereafter, the group formatter 303 may insert the  $1/2$ -rate encoded mobile service data to region C/D. In another example, the block processor 302 may encode the mobile service data, which are to be inserted in region C/D, at a coding rate having higher error correction ability than the  $1/4$ -coding rate. Then, the group formatter 303 may either insert the encoded mobile service data to region C/D, as described above, or leave the data in a reserved region for future usage.

**[00262]** According to another embodiment of the present invention, the block processor 302 may perform a 1/H-rate encoding process in SCCC block units. Herein, the SCCC block includes at least one M/H block.

**[00263]** At this point, when 1/H-rate encoding is performed in M/H block units, the M/H blocks (B1 to B10) and the SCCC block (SCB1 to SCB10) become identical to one another (*i.e.*, SCB1=B1, SCB2=B2, SCB3=B3, SCB4=B4, SCB5=B5, SCB6=B6, SCB7=B7, SCB8=B8, SCB9=B9, and SCB10=B10). For example, the M/H block 1 (B1) may be encoded at the coding rate of 1/2, the M/H block 2 (B2) may be encoded at the coding rate of 1/4, and the M/H block 3 (B3) may be encoded at the coding rate of 1/2. The coding rates are applied respectively to the remaining M/H blocks.

**[00264]** Alternatively, a plurality of M/H blocks within regions A, B, C, and D may be grouped into one SCCC block, thereby being encoded at a coding rate of 1/H in SCCC block units. Accordingly, the receiving performance of region C/D may be enhanced. For example, M/H block 1 (B1) to M/H block 5 (B5) may be grouped into one SCCC block and then encoded at a coding rate of 1/2. Thereafter, the group formatter 303 may insert the 1/2-rate encoded mobile service data to a section starting from M/H block 1 (B1) to M/H block 5 (B5).

**[00265]** Furthermore, M/H block 6 (B6) to M/H block 10 (B10) may be grouped into one SCCC block and then encoded at a coding rate of 1/4. Thereafter, the group formatter 303 may insert the 1/4-rate encoded mobile service data to another section starting from M/H block 6 (B6) to M/H block 10 (B10). In this case, one data group may consist of two SCCC blocks.

**[00266]** According to another embodiment of the present invention, one SCCC block may be formed by grouping two M/H blocks. For example, M/H block 1 (B1) and M/H block 6 (B6) may be grouped into one SCCC block (SCB1). Similarly, M/H block 2 (B2) and M/H block 7 (B7) may be grouped into another SCCC

block (SCB2). Also, M/H block 3 (B3) and M/H block 8 (B8) may be grouped into another SCCC block (SCB3). And, M/H block 4 (B4) and M/H block 9 (B9) may be grouped into another SCCC block (SCB4). Furthermore, M/H block 5 (B5) and M/H block 10 (B10) may be grouped into another SCCC block (SCB5). In the above-described example, the data group may consist of 10 M/H blocks and 5 SCCC blocks. Accordingly, in a data (or signal) receiving environment undergoing frequent and severe channel changes, the receiving performance of regions C and D, which is relatively more deteriorated than the receiving performance of region A, may be reinforced. Furthermore, since the number of mobile service data symbols increases more and more from region A to region D, the error correction encoding performance becomes more and more deteriorated. Therefore, when grouping a plurality of M/H block to form one SCCC block, such deterioration in the error correction encoding performance may be reduced.

[00267] As described-above, when the block processor 302 performs encoding at a 1/H-coding rate, information associated with SCCC should be transmitted to the receiving system in order to accurately recover the mobile service data.

[00268] Table 7 below shows an example of a SCCC block mode, which indicating the relation between an M/H block and an SCCC block, among diverse SCCC block information.

[00269] Table 7

| SCCC<br>Block<br>Mode | 00                                    | 01                                     | 10           | 11           |
|-----------------------|---------------------------------------|----------------------------------------|--------------|--------------|
| Desc<br>ription       | One M/H<br>Block<br>per SCCC<br>Block | Two M/H<br>Blocks<br>per SCCC<br>Block | Rese<br>rved | Reser<br>ved |
| SCB                   | SCB input,<br>M/H Block               | SCB input,<br>M/H Blocks               |              |              |

|           |     |          |  |  |
|-----------|-----|----------|--|--|
| SCB1      | B1  | B1 + B6  |  |  |
| SCB2      | B2  | B2 + B7  |  |  |
| SCB3      | B3  | B3 + B8  |  |  |
| SCB4      | B4  | B4 + B9  |  |  |
| SCB5      | B5  | B5 + B10 |  |  |
| SCB6      | B6  | -        |  |  |
| SCB7      | B7  | -        |  |  |
| SCB8      | B8  | -        |  |  |
| SCB9      | B9  | -        |  |  |
| SCB1<br>0 | B10 | -        |  |  |

**[00270]** More specifically, Table 4 shows an example of 2 bits being allocated in order to indicate the SCCC block mode. For example, when the SCCC block mode value is equal to '00', this indicates that the SCCC block and the M/H block are identical to one another. Also, when the SCCC block mode value is equal to '01', this indicates that each SCCC block is configured of 2 M/H blocks.

**[00271]** As described above, if one data group is configured of 2 SCCC blocks, although it is not indicated in Table 7, this information may also be indicated as the SCCC block mode. For example, when the SCCC block mode value is equal to '10', this indicates that each SCCC block is configured of 5 M/H blocks and that one data group is configured of 2 SCCC blocks. Herein, the number of M/H blocks included in an SCCC block and the position of each M/H block may vary depending upon the settings made by the system designer. Therefore, the present invention will not be limited to the examples given herein. Accordingly, the SCCC mode information may also be expanded.

**[00272]** An example of a coding rate information of the SCCC block, i.e., SCCC outer code mode, is shown in Table 8 below.

**[00273]** Table 8

| SCCC outer<br>code mode (2 bits) | Description                                  |
|----------------------------------|----------------------------------------------|
| 00                               | Outer code rate of SCCC block<br>is 1/2 rate |
| 01                               | Outer code rate of SCCC block<br>is 1/4 rate |
| 10                               | Reserved                                     |
| 11                               | Reserved                                     |

**[00274]** More specifically, Table 8 shows an example of 2 bits being allocated in order to indicate the coding rate information of the SCCC block. For example, when the SCCC outer code mode value is equal to '00', this indicates that the coding rate of the corresponding SCCC block is 1/2. And, when the SCCC outer code mode value is equal to '01', this indicates that the coding rate of the corresponding SCCC block is 1/4.

**[00275]** If the SCCC block mode value of Table 7 indicates '00', the SCCC outer code mode may indicate the coding rate of each M/H block with respect to each M/H block. In this case, since it is assumed that one data group includes 10 M/H blocks and that 2 bits are allocated for each SCCC block mode, a total of 20 bits are required for indicating the SCCC block modes of the 10 M/H modes.

**[00276]** In another example, when the SCCC block mode value of Table 7 indicates '00', the SCCC outer code mode may indicate the coding rate of each region with respect to each region within the data group. In this case, since it is assumed that one data group includes 4 regions (i.e., regions A, B, C, and D) and that 2 bits are allocated for each SCCC block mode, a total of 8 bits are required for indicating the SCCC block modes of the 4 regions.

**[00277]** In another example, when the SCCC block mode value of Table 7 is equal to '01', each of the regions A, B, C, and D within the data group has the same SCCC outer code mode.

**[00278]** Meanwhile, an example of an SCCC output block length (SOBL) for each SCCC block, when the SCCC block mode value is equal to '00', is shown in Table 9 below.

**[00279]** Table 9

| SCCC Block  | SOBL | SIBL        |             |
|-------------|------|-------------|-------------|
|             |      | 1/2<br>rate | 1/4<br>rate |
| SCB1 (B1)   | 528  | 264         | 132         |
| SCB2 (B2)   | 1536 | 768         | 384         |
| SCB3 (B3)   | 2376 | 1188        | 594         |
| SCB4 (B4)   | 2388 | 1194        | 597         |
| SCB5 (B5)   | 2772 | 1386        | 693         |
| SCB6 (B6)   | 2472 | 1236        | 618         |
| SCB7 (B7)   | 2772 | 1386        | 693         |
| SCB8 (B8)   | 2508 | 1254        | 627         |
| SCB9 (B9)   | 1416 | 708         | 354         |
| SCB10 (B10) | 480  | 240         | 120         |

**[00280]** More specifically, when given the SCCC output block length (SOBL) for each SCCC block, an SCCC input block length (SIBL) for each corresponding SCCC block may be decided based upon the outer coding rate of each SCCC block. The SOBL is equivalent to the number of SCCC output (or outer-encoded) bytes for each SCCC block. And, the SIBL is equivalent to the number of SCCC input (or payload) bytes for each SCCC block.

**[00281]** Table 10 below shows an example of the SOBL and SIBL for each SCCC block, when the SCCC block mode value is equal to '01'.

**[00282]** Table 10

| SCCC Block    | SOBL | SIBL        |             |
|---------------|------|-------------|-------------|
|               |      | 1/2<br>rate | 1/4<br>rate |
| SCB1 (B1+B6)  | 528  | 264         | 132         |
| SCB2 (B2+B7)  | 1536 | 768         | 384         |
| SCB3 (B3+B8)  | 2376 | 1188        | 594         |
| SCB4 (B4+B9)  | 2388 | 1194        | 597         |
| SCB5 (B5+B10) | 2772 | 1386        | 693         |

**[00283]** In order to do so, as shown in FIG. 22, the block processor 302 includes a RS frame portion-SCCC block converter 511, a byte-bit converter 512, a convolution encoder 513, a symbol interleaver 514, a symbol-byte converter 515, and an SCCC block-M/H block converter 516.

**[00284]** The convolutional encoder 513 and the symbol interleaver 514 are virtually concatenated with the trellis encoding module in the post-processor in order to configure an SCCC block.

**[00285]** More specifically, the RS frame portion-SCCC block converter 511 divides the RS frame portions, which are being inputted, into multiple SCCC blocks using the SIBL of Table 9 and Table 10 based upon the RS code mode, SCCC block mode, and SCCC outer code mode. Herein, the M/H frame encoder 301 may output only primary RS frame portions or both primary RS frame portions and secondary RS frame portions in accordance with the RS frame mode.

**[00286]** When the RS Frame mode is set to '00', a portion of the primary RS Frame equal to the amount of data, which are to be SCCC outer encoded and mapped to 10 M/H blocks (B1 to B10) of a data group, will be provided to the block processor 302. When the SCCC block mode value is equal to '00', then the primary RS frame portion will be split into 10 SCCC Blocks according to Table 9. Alternatively, when the SCCC block mode

value is equal to '01', then the primary RS frame will be split into 5 SCCC blocks according to Table 10. When the RS frame mode value is equal to '01', then the block processor 302 may receive two RS frame portions. The RS frame mode value of '01' will not be used with the SCCC block mode value of '01'. The first portion from the primary RS frame will be SCCC-outer-encoded as SCCC Blocks SCB3, SCB4, SCB5, SCB6, SCB7, and SCB8 by the block processor 302. The SCCC Blocks SCB3 and SCB8 will be mapped to region B and the SCCC blocks SCB4, SCB5, SCB6, and SCB7 shall be mapped to region A by the group formatter 303. The second portion from the secondary RS frame will also be SCCC-outer-encoded, as SCB1, SCB2, SCB9, and SCB10, by the block processor 302. The group formatter 303 will map the SCCC blocks SCB1 and SCB10 to region D as the M/H blocks B1 and B10, respectively. Similarly, the SCCC blocks SCB2 and SCB9 will be mapped to region C as the M/H blocks B2 and B9.

**[00287]** The byte-bit converter 512 identifies the mobile service data bytes of each SCCC block outputted from the RS frame portion-SCCC block converter 511 as data bits, which are then outputted to the convolution encoder 513.

**[00288]** The convolution encoder 513 performs one of 1/2-rate encoding and 1/4-rate encoding on the inputted mobile service data bits.

**[00289]** FIG. 23 illustrates a detailed block diagram of the convolution encoder 513. The convolution encoder 513 includes two delay units 521 and 523 and three exclusive-or gates 522, 524, and 525. Herein, the convolution encoder 513 encodes an input data bit U and outputs the coded bit U to 5 bits (u0 to u4).

**[00290]** At this point, the input data bit U is directly outputted as uppermost bit u0 and simultaneously encoded as lower bit u1u2u3u4 and then outputted. More specifically, the input data bit U is directly outputted as the uppermost bit u0

and simultaneously outputted to the first and third exclusive-or gates 522 and 525. The first exclusive-or gate 522 performs exclusive-or operation on the input data bit U and the output bit of the first delay unit 521 and, then, outputs to the second delay unit 523. Then, the data bit delayed by a pre-determined time (e.g., by 1 clock) in the second delay unit 523 is outputted as a lower bit u1 and simultaneously fed-back to the first delay unit 521. The first delay unit 521 delays the data bit fed-back from the second delay unit 523 by a pre-determined time (e.g., by 1 clock). Then, the first delay unit 521 outputs the delayed data bit as a lower bit u2 and, at the same time, outputs the fed-back data to the first exclusive-or gate 522 and the second exclusive-or gate 524.

**[00291]** The second exclusive-or gate 524 performs exclusive-or operation on the data bits outputted from the first and second delay units 521 and 523 and outputs as a lower bit u3. The third exclusive-or gate 525 performs on exclusive-or operation on the input data bit U and the output of the second delay unit 523 and outputs as a lower bit u4.

**[00292]** At this point, the first and second delay units 521 and 523 are reset to '0', at the starting point of each SCCC block. The convolution encoder 513 of FIG. 23 may be used as a 1/2-rate encoder or a 1/4-rate encoder.

**[00293]** More specifically, when a portion of the output bit of the convolution encoder 513, shown in FIG. 23, is selected and outputted, the convolution encoder 513 may be used as one of a 1/2-rate encoder and a 1/4-rate encoder.

**[00294]** Table 11 below shown an example of output symbols of the convolution encoder 513.

**[00295]** Table 11

| Reg<br>ion | 1/2<br>rate | 1/4 rate                  |                           |
|------------|-------------|---------------------------|---------------------------|
|            |             | SCCC block mode<br>= '00' | SCCC block mode<br>= '01' |

|    |       |                |                |
|----|-------|----------------|----------------|
| A, | (u0   | (u0, u2), (u1, | (u0, u2), (u1, |
| B  | , u1) | u3)            | u4)            |
| C, |       | (u0, u1), (u3, |                |
| D  |       | u4)            |                |

**[00296]** For example, at the 1/2-coding rate, 1 output symbol (i.e., u0 and u1 bits) may be selected and outputted. And, at the 1/4-coding rate, depending upon the SCCC block mode, 2 output symbols (i.e., 4 bits) may be selected and outputted. For example, when the SCCC block mode value is equal to '01', and when an output symbol configured of u0 and u2 and another output symbol configured of u1 and u4 are selected and outputted, a 1/4-rate coding result may be obtained.

**[00297]** The mobile service data encoded at the coding rate of 1/2 or 1/4 by the convolution encoder 513 are outputted to the symbol interleaver 514.

**[00298]** The symbol interleaver 514 performs block interleaving, in symbol units, on the output data symbol of the convolution encoder 513. More specifically, the symbol interleaver 514 is a type of block interleaver. Any interleaver performing structural rearrangement (or realignment) may be applied as the symbol interleaver 514 of the block processor. However, in the present invention, a variable length symbol interleaver that can be applied even when a plurality of lengths is provided for the symbol, so that its order may be rearranged, may also be used.

**[00299]** FIG. 24 illustrates a symbol interleaver according to an embodiment of the present invention. Particularly, FIG. 30 illustrates an example of the symbol interleaver when B=2112 and L=4096.

**[00300]** Herein, B indicates a block length in symbols that are outputted for symbol interleaving from the convolution encoder 513. And, L represents a block length in symbols that are actually interleaved by the symbol interleaver 514. At

this point, the block length in symbols  $B$  inputted to the symbol interleaver 514 is equivalent to  $4 \times SOBL$ . More specifically, since one symbol is configured of 2 bits, the value of  $B$  may be set to be equal to  $4 \times SOBL$ .

**[00301]** In the present invention, when performing the symbol-interleaving process, the conditions of  $L=2^m$  (wherein  $m$  is an integer) and of  $L \geq B$  should be satisfied. If there is a difference in value between  $B$  and  $L$ ,  $(L-B)$  number of null (or dummy) symbols is added, thereby creating an interleaving pattern, as shown in  $P'(i)$  of FIG. 24.

**[00302]** Therefore,  $B$  becomes a block size of the actual symbols that are inputted to the symbol interleaver 514 in order to be interleaved.  $L$  becomes an interleaving unit when the interleaving process is performed by an interleaving pattern created from the symbol interleaver 514.

**[00303]** Equation 5 shown below describes the process of sequentially receiving  $B$  number of symbols, the order of which is to be rearranged, and obtaining an  $L$  value satisfying the conditions of  $L=2^m$  (wherein  $m$  is an integer) and of  $L \geq B$ , thereby creating the interleaving so as to realign (or rearrange) the symbol order.

**[00304]** Equation 5

In relation to all places, wherein  $0 \leq i \leq B-1$ ,

$$P'(i) = \{89 \times i \times (i+1) / 2\} \bmod L$$

Herein,  $L \geq B$ ,  $L=2^m$ , wherein  $m$  is an integer.

**[00305]** As shown in  $P'(i)$  of FIG. 24 and Equation 5, the order of  $B$  number of input symbols and  $(L-B)$  number of null symbols is rearranged by using the above-mentioned Equation 6. Then, as shown in  $P(i)$  of FIG. 24, the null byte places are removed, so as to rearrange the order. Starting with the lowest value of  $i$ , the  $P(i)$  are shifted to the left in order to fill the empty entry locations. Thereafter, the symbols of

the aligned interleaving pattern  $P(i)$  are outputted to the symbol-byte converter 515 in order.

**[00306]** Herein, the symbol-byte converter 515 converts to bytes the mobile service data symbols, having the rearranging of the symbol order completed and then outputted in accordance with the rearranged order, and thereafter outputs the converted bytes to the SCCC block-M/H block converter 516. The SCCC block-M/H block converter 516 converts the symbol-interleaved SCCC blocks to M/H blocks, which are then outputted to the group formatter 303.

**[00307]** If the SCCC block mode value is equal to '00', the SCCC block is mapped at a one-to-one (1:1) correspondence with each M/H block within the data group. In another example, if the SCCC block mode value is equal to '01', each SCCC block is mapped with two M/H blocks within the data group. For example, the SCCC block SCB1 is mapped with (B1, B6), the SCCC block SCB2 is mapped with (B2, B7), the SCCC block SCB3 is mapped with (B3, B8), the SCCC block SCB4 is mapped with (B4, B9), and the SCCC block SCB5 is mapped with (B5, B10). The M/H block that is outputted from the SCCC block-M/H block converter 516 is configured of mobile service data and FEC redundancy. In the present invention, the mobile service data as well as the FEC redundancy of the M/H block will be collectively considered as mobile service data.

**[00308]**

**[00309]** Group Formatter

**[00310]** The group formatter 303 inserts data of M/H blocks outputted from the block processor 302 to the corresponding M/H blocks within the data group, which is formed in accordance with a pre-defined rule. Also, in association with the data-deinterleaving process, the group formatter 303 inserts various place holders (or known data place holders) in the corresponding region within the data group.

**[00311]** More specifically, apart from the encoded mobile service data outputted from the block processor 302, the group formatter 303 also inserts MPEG header place holders, non-systematic RS parity place holders, main service data place holders, which are associated with the data deinterleaving in a later process, as shown in FIG. 5. Herein, the main service data place holders are inserted because the mobile service data bytes and the main service data bytes are alternately mixed with one another in regions B to D based upon the input of the data deinterleaver, as shown in FIG. 5. For example, based upon the data outputted after data deinterleaving, the place holder for the MPEG header may be allocated at the very beginning of each packet. Also, in order to configure an intended group format, dummy bytes may also be inserted. Furthermore, the group formatter 303 inserts initialization data (i.e., trellis initialization byte) of the trellis encoding module 256 in the corresponding regions. For example, the initialization data may be inserted in the beginning of the known data sequence. The initialization data is used for initializing memories within the trellis encoding module 256, and is not transmitted to the receiving system.

**[00312]** Additionally, the group formatter 303 may also insert signaling information, which are encoded and outputted from the signaling encoder 304, in corresponding regions within the data group.

**[00313]** At this point, reference may be made to the signaling information when the group formatter 303 inserts each data type and respective place holders in the data group. The process of encoding the signaling information and inserting the encoded signaling information to the data group will be described in detail in a later process.

**[00314]** After inserting each data type and respective place holders in the data group, the group formatter 303 may deinterleave the data and respective place holders, which have

been inserted in the data group, as an inverse process of the data interleaver, thereby outputting the deinterleaved data and respective place holders to the packet formatter 305. The group formatter 303 may include a group format organizer 527, and a data deinterleaver 529, as shown in FIG. 25. The group format organizer 527 inserts data and respective place holders in the corresponding regions within the data group, as described above. And, the data deinterleaver 529 deinterleaves the inserted data and respective place holders as an inverse process of the data interleaver.

**[00315]** The packet formatter 305 removes the main service data place holders and the RS parity place holders and replaces the MPEG header place holders with MPEG header that were allocated for the deinterleaving process from the deinterleaved data being inputted.

**[00316]** Also, when the group formatter 303 inserts known data place holders, the packet formatter 303 may insert actual known data in the known data place holders, or may directly output the known data place holders without any modification in order to make replacement insertion in a later process.

**[00317]** Thereafter, the packet formatter 305 identifies the data within the packet-formatted data group, as described above, as a 188-byte unit mobile service data packet (*i.e.*, MPEG TS packet), which is then provided to the packet multiplexer 240.

**[00318]** The packet multiplexer 240 multiplexes the mobile service data packets packet-formatted and outputted from the packet formatter 306 and main service data packets outputted from the packet jitter mitigator 220. Then, the packet multiplexer 240 outputs the multiplexed data packets to the data randomizer 251 of the post-processor 250. If the packet multiplexer 240 receives 118 mobile service data packets from the packet formatter 305, 37 mobile service data packets are placed before a place for inserting VSB field synchronization.

Then, the remaining 81 mobile service data packets are placed after the place for inserting VSB field synchronization. The multiplexing method may be adjusted by diverse variables of the system design. The multiplexing method and multiplexing rule of the packet multiplexer 240 will be described in more detail in a later process.

[00319] Also, since a data group including mobile service data in-between the data bytes of the main service data is multiplexed (or allocated) during the packet multiplexing process, the shifting of the chronological position (or place) of the main service data packet becomes relative. Also, a system object decoder (*i.e.*, MPEG decoder) for processing the main service data of the receiving system, receives and decodes only the main service data and recognizes the mobile service data packet as a null data packet.

[00320] Therefore, when the system object decoder of the receiving system receives a main service data packet that is multiplexed with the data group, a packet jitter occurs.

[00321] At this point, since a multiple-level buffer for the video data exists in the system object decoder and the size of the buffer is relatively large, the packet jitter generated from the packet multiplexer 240 does not cause any serious problem in case of the video data. However, since the size of the buffer for the audio data in the object decoder is relatively small, the packet jitter may cause considerable problem.

[00322] More specifically, due to the packet jitter, an overflow or underflow may occur in the buffer for the main service data of the receiving system (*e.g.*, the buffer for the audio data).

[00323] Therefore, the packet jitter mitigator 220 re-adjusts the relative position of the main service data packet so that the overflow or underflow does not occur in the system object decoder.

**[00324]** In the present invention, examples of repositioning places for the audio data packets within the main service data in order to minimize the influence on the operations of the audio buffer will be described in detail. The packet jitter mitigator 220 repositions the audio data packets in the main service data section so that the audio data packets of the main service data can be as equally and uniformly aligned and positioned as possible.

**[00325]** Additionally, when the positions of the main service data packets are relatively re-adjusted, associated program clock reference (PCR) values may also be modified accordingly. The PCR value corresponds to a time reference value for synchronizing the time of the MPEG decoder. Herein, the PCR value is inserted in a specific region of a TS packet and then transmitted. In the example of the present invention, the packet jitter mitigator 220 also performs the operation of modifying the PCR value.

**[00326]** The output of the packet jitter mitigator 220 is inputted to the packet multiplexer 240. As described above, the packet multiplexer 240 multiplexes the main service data packets outputted from the packet jitter mitigator 220 with the mobile service data packets outputted from the pre-processor 230 into a burst structure in accordance with a pre-determined multiplexing rule. Then, the packet multiplexer 240 outputs the multiplexed data packets to the data randomizer 251 of the post-processor 250.

**[00327]** If the inputted data correspond to the main service data packet, the data randomizer 251 performs the same randomizing process as that of the conventional randomizer and outputs to the RS encoder/non-systematic RS encoder 252. More specifically, all data bytes of the main service data packet are randomized by using a pseudo random byte generated from the data randomizer 251. Thereafter, the randomized data are outputted to the RS encoder/non-systematic RS encoder 252. On

the other hand, if the inputted data correspond to the mobile service data packet, the data randomizer 251 performs randomizing only MPEG header byte of the mobile service data packet and outputs to the RS encoder/non-systematic RS encoder 252. The randomizing of the mobile service data included in the mobile service data packet was performed by the data randomizer 411 of the primary encoder 410 and/or the data randomizer 422 of the secondary encoder 420.

**[00328]** The RS encoder/non-systematic RS encoder 252 performs an RS encoding process on the data being randomized by the data randomizer 251 or on the data bypassing the data randomizer 251, so as to add 20 bytes of RS parity data. Thereafter, the processed data are outputted to the data interleaver 253. Herein, if the inputted data correspond to the main service data packet, the RS encoder/non-systematic RS encoder 252 performs the same systematic RS encoding process as that of the conventional broadcasting system, thereby adding the 20-byte RS parity data at the end of the 187-byte data. Alternatively, if the inputted data correspond to the mobile service data packet, the RS encoder/non-systematic RS encoder 252 performs a non-systematic RS encoding process. At this point, the 20-byte RS parity data obtained from the non-systematic RS encoding process are inserted in a pre-decided parity byte place within the mobile service data packet.

**[00329]** The data interleaver 253 corresponds to a byte unit convolutional interleaver.

**[00330]** The output of the data interleaver 253 is inputted to the parity replacer 254 and to the non-systematic RS encoder 255.

**[00331]** Meanwhile, a process of initializing a memory within the trellis encoding module 256 is primarily required in order to decide the output data of the trellis encoding module 256, which is located after the parity replacer 254, as the known data pre-defined according to an agreement between the

receiving system and the transmitting system. More specifically, the memory of the trellis encoding module 256 should first be initialized before the received known data sequence is trellis-encoded.

**[00332]** At this point, the beginning portion of the known data sequence that is received corresponds to the initialization data (i.e., trellis initialization data bytes) and not to the actual known data. Herein, the initialization data has been included in the data by the group formatter within the pre-processor 230 in an earlier process. Therefore, the process of replacing the initialization data with memory values within the trellis encoding module 256 are required to be performed immediately before the inputted known data sequence is trellis-encoded.

**[00333]** More specifically, the initialization data are replaced with the memory value within the trellis encoding module 256, thereby being inputted to the trellis encoding module 256. At this point, the memory value replacing the initialization data are process with (or calculated by) an exclusive OR (XOR) operation with the respective memory value within the trellis encoding module 256, so as to be inputted to the corresponding memory. Therefore, the corresponding memory is initialized to '0'. Additionally, a process of using the memory value replacing the initialization data to re-calculate the RS parity, so that the re-calculated RS parity value can replace the RS parity being outputted from the data interleaver 253, is also required.

**[00334]** Therefore, the non-systematic RS encoder 255 receives the mobile service data packet including the initialization data from the data interleaver 253 and also receives the memory value from the trellis encoding module 256. Among the inputted mobile service data packet, the initialization data are replaced with the memory value, and the RS parity data that are added to the mobile service data

packet are removed and processed with non-systematic RS encoding. Thereafter, the new RS parity obtained by performing the non-systematic RS encoding process is outputted to the parity replacer 255. Accordingly, the parity replacer 255 selects the output of the data interleaver 253 as the data within the mobile service data packet, and the parity replacer 255 selects the output of the non-systematic RS encoder 255 as the RS parity. The selected data are then outputted to the trellis encoding module 256.

**[00335]** Meanwhile, if the main service data packet is inputted or if the mobile service data packet, which does not include any initialization data that are to be replaced, is inputted, the parity replacer 254 selects the data and RS parity that are outputted from the data interleaver 253. Then, the parity replacer 254 directly outputs the selected data to the trellis encoding module 256 without any modification.

**[00336]** The trellis encoding module 256 converts the byte-unit data to symbol units and performs a 12-way interleaving process so as to trellis-encode the received data. Thereafter, the processed data are outputted to the synchronization multiplexer 260.

**[00337]** FIG. 26 illustrates a detailed diagram of one of 12 trellis encoders included in the trellis encoding module 256. Herein, the trellis encoder includes first and second multiplexers 531 and 541, first and second exclusive OR (XOR) gates 532 and 542, and first to third memories 533, 542, and 544.

**[00338]** More specifically, the first to third memories 533, 542, and 544 are initialized by the memory value instead of the initialization data from the parity replacer 254. More specifically, when the first symbol (i.e., two bits), which are converted from initialization data (i.e., each trellis initialization data byte), are inputted, the input bits of the

trellis encoder will be replaced by the memory values of the trellis encoder, as shown in FIG. 26.

**[00339]** Since 2 symbols (*i.e.*, 4 bits) are required for trellis initialization, the last 2 symbols (*i.e.*, 4 bits) from the trellis initialization bytes are not used for trellis initialization and are considered as a symbol from a known data byte and processed accordingly.

**[00340]** When the trellis encoder of FIG. 26 is in the initialization mode, the input comes from an internal trellis status (or state) and not from the parity replacer 254. When the trellis encoder is in the normal mode, the input symbol (X2X1) provided from the parity replacer 254 will be processed. The trellis encoder provides the converted (or modified) input data for trellis initialization to the non-systematic RS encoder 255.

**[00341]** More specifically, when a selection signal designates a normal mode, the first multiplexer 531 selects an upper bit X2 of the input symbol. And, when a selection signal designates an initialization mode, the first multiplexer 531 selects the output of the first memory 533 and outputs the selected output data to the first XOR gate 532. The first XOR gate 532 performs XOR operation on the output of the first multiplexer 531 and the output of the first memory 533, thereby outputting the added result to the first memory 533 and, at the same time, as a most significant (or uppermost) bit Z2. The first memory 533 delays the output data of the first XOR gate 532 by 1 clock, thereby outputting the delayed data to the first multiplexer 531 and the first XOR gate 532. Meanwhile, when a selection signal designates a normal mode, the second multiplexer 541 selects a lower bit X1 of the input symbol. And, when a selection signal designates an initialization mode, the second multiplexer 541 selects the output of the second memory 542, thereby outputting the selected result to the second XOR gate 543 and, at the same

time, as a lower bit Z1. The second XOR gate 543 performs XOR operation on the output of the second multiplexer 541 and the output of the second memory 542, thereby outputting the added result to the third memory 544. The third memory 544 delays the output data of the second XOR gate 543 by 1 clock, thereby outputting the delayed data to the second memory 542 and, at the same time, as a least significant (or lowermost) bit Z0. The second memory 542 delays the output data of the third memory 544 by 1 clock, thereby outputting the delayed data to the second XOR gate 543 and the second multiplexer 541.

**[00342]** The select signal designates an initialization mode during the first two symbols that are converted from the initialization data.

**[00343]** For example, when the select signal designates an initialization mode, the first XOR gate 532 performs an XOR operation on the value of the first memory 533, which is provided through the first multiplexer 531, and on a memory value that is directly provided from the first memory 533. That is, the first XOR gate 532 performs an XOR operation on 2 bits having the same value. Generally, when only one of the two bits belonging to the operand is '1', the result of the XOR gate is equal to '1'. Otherwise, the result of the XOR gate becomes equal to '0'. Therefore, when the value of the first memory 533 is processed with an XOR operation, the result is always equal to '0'. Furthermore, since the output of the first XOR gate 532, i.e., '0', is inputted to the first memory 533, the first memory 533 is initialized to '0'.

**[00344]** Similarly, when the select signal designates an initialization mode, the second XOR gate 543 performs an XOR operation on the value of the second memory 542, which is provided through the second multiplexer 541, and on a memory value that is directly provided from the second memory 542. Therefore, the output of the second XOR gate 543 is also always equal to '0'. Since the output of the second XOR gate

543, i.e., '0', is inputted to the third memory 544, the third memory 544 is also initialized to '0'. The output of the third memory 544 is inputted to the second memory 542 in the next clock, thereby initializing the second memory 542 to '0'. In this case also, the select signal designates the initialization mode.

**[00345]** More specifically, when the first symbol being converted from the initialization data byte replaces the values of the first memory 533 and the second memory 542, thereby being inputted to the trellis encoder, each of the first and third memories 533 and 544 within the trellis encoder is initialized to '00'. Following the process, when the second symbol being converted from the initialization data byte replaces the values of the first memory 533 and the second memory 542, thereby being inputted to the trellis encoder, each of the first, second, and third memories 533, 542, and 544 within the trellis encoder is initialized to '000'. As described above, 2 symbols are required to initialize the memory of the trellis encoder. At this point, while the select signal designates an initialization mode, the output bits (X2'X1') of the first and second memories 533 and 542 are inputted to the non-systematic RS encoder 255, so as to perform a new RS parity calculation process.

**[00346]** The synchronization multiplexer 260 inserts a field synchronization signal and a segment synchronization signal to the data outputted from the trellis encoding module 256 and, then, outputs the processed data to the pilot inserter 271 of the transmission unit 270.

**[00347]** Herein, the data having a pilot inserted therein by the pilot inserter 271 are modulated by the modulator 272 in accordance with a pre-determined modulating method (e.g., a VSB method). Thereafter, the modulated data are transmitted to each receiving system through the radio frequency (RF) up-converter 273.

[00348]

[00349]        Assignment of Known Data (or Training Signal)

[00350]        The transmission system inserts long and regularly spaced training sequences (i.e., known data sequences) into each data group. Each data group contains 6 training sequences. The training sequences are specified before trellis-encoding. The training sequences are then trellis-encoded and these trellis-encoded sequences also are known sequences. This is because the trellis encoder memories are initialized to pre-determined values at the beginning of each sequence. The form of the 6 training sequences at the byte level (before trellis-encoding) is shown in FIG. 27. FIG. 27 is an embodiment of the arrangement of the training sequences that is performed by the group formatter 303.

[00351]        The 1<sup>st</sup> training sequence is located at the last 2 segments of the 3<sup>rd</sup> M/H block (B3). The 2<sup>nd</sup> training sequence may be inserted at the 2<sup>nd</sup> and 3<sup>rd</sup> segments of the 4<sup>th</sup> M/H block (B4). The 2<sup>nd</sup> training sequence is next to the signaling area, as shown in FIG. 5. Then, the 3<sup>rd</sup> training sequence, the 4<sup>th</sup> training sequence, the 5<sup>th</sup> training sequence, and the 6<sup>th</sup> training sequence may be placed at the last 2 segments of the 4<sup>th</sup>, 5<sup>th</sup>, 6<sup>th</sup>, and 7<sup>th</sup> M/H blocks (B4, B5, B6, and B7), respectively.

[00352]        As shown in FIG. 27, the 1<sup>st</sup> training sequence, the 3<sup>rd</sup> training sequence, the 4<sup>th</sup> training sequence, the 5<sup>th</sup> training sequence, and the 6<sup>th</sup> training sequence are spaced 16 segments apart from one another. Referring to FIG. 27, the dotted area indicates trellis initialization data bytes, the lined area indicates training data bytes, and the white area includes other bytes such as the FEC-coded M/H service data bytes, FEC-coded signaling data, main service data bytes, RS parity data bytes (for backwards compatibility with legacy ATSC receivers) and/or dummy data bytes.

**[00353]** FIG. 28 illustrates the training sequences (at the symbol level) after trellis-encoding by the trellis encoder. Referring to FIG. 28, the dotted area indicates data segment sync symbols, the lined area indicates training data symbols, and the white area includes other symbols, such as FEC-coded mobile service data symbols, FEC-coded signaling data symbols, main service data symbols, RS parity data symbols, dummy data symbols, trellis initialization data symbols, and/or the first part of the training sequence data symbols.

**[00354]** After the trellis-encoding process, the last 1416 (=588+828) symbols of the 1<sup>st</sup> training sequence, the 3<sup>rd</sup> training sequence, the 4<sup>th</sup> training sequence, the 5<sup>th</sup> training sequence, and the 6<sup>th</sup> training sequence commonly share the same data pattern. The 2<sup>nd</sup> training sequence has a first 528-symbol sequence and a second 528-symbol sequence that have the same data pattern.

**[00355]** More specifically, the 528-symbol sequence is repeated after the 4-symbol data segment synchronization signal. At the end of each training sequence, the memory contents of the twelve modified trellis encoders shall be set to zero(0).

**[00356]**

**[00357]** Processing Signaling Information

**[00358]** The present invention assigns signaling information areas for inserting signaling information to some areas within each data group.

**[00359]** FIG. 29 illustrates an example of assigning signaling information areas for inserting signaling information starting from the 1<sup>st</sup> segment of the 4<sup>th</sup> M/H block (B4) to a portion of the 2<sup>nd</sup> segment. More specifically, 276(=207+69) bytes of the 4<sup>th</sup> M/H block (B4) in each data group are assigned as the signaling information area. In other words, the signaling information area consists of 207 bytes of the 1<sup>st</sup> segment and the first 69 bytes of the 2<sup>nd</sup> segment of the 4<sup>th</sup> M/H

block (B4). For example, the 1<sup>st</sup> segment of the 4<sup>th</sup> M/H block (B4) corresponds to the 17<sup>th</sup> or 173<sup>rd</sup> segment of a VSB field.

**[00360]** For example, when the data group includes 6 known data sequences, as shown in FIG. 27 and FIG. 28, the signaling information area is located between the first known data sequence and the second known data sequence. More specifically, the first known data sequence is inserted in the last 2 segments of the 3<sup>rd</sup> M/H block (B3), and the second known data sequence is inserted in the 2<sup>nd</sup> and 3<sup>rd</sup> segments of the 4<sup>th</sup> M/H block (B4). Furthermore, the 3<sup>rd</sup> to 6<sup>th</sup> known data sequences are respectively inserted in the last 2 segments of each of the 4<sup>th</sup>, 5<sup>th</sup>, 6<sup>th</sup>, and 7<sup>th</sup> M/H blocks (B4, B5, B6, and B7). The 1<sup>st</sup> and 3<sup>rd</sup> to 6<sup>th</sup> known data sequences are spaced apart by 16 segments.

**[00361]** The signaling information that is to be inserted in the signaling information area is FEC-encoded by the signaling encoder 304, thereby inputted to the group formatter 303.

**[00362]** The group formatter 303 inserts the signaling information, which is FEC-encoded and outputted by the signaling encoder 304, in the signaling information area within the data group.

**[00363]** Herein, the signaling information may be identified by two different types of signaling channels: a transmission parameter channel (TPC) and a fast information channel (FIC).

**[00364]** Herein, the TPC data is transmitted through the TPC and corresponds to signaling information including transmission parameters, such as RS frame information, RS encoding information, FIC information, data group information, SCCC information, and M/H frame information and so on. However, the TPC data presented herein is merely exemplary. And, since the adding or deleting of signaling information included in the TPC may be easily adjusted and modified by one skilled in the art, the present invention will, therefore, not be limited to the examples set forth herein. Also, the TPC data includes

parameters that are mostly used in a physical layer module. And, since the TPC data are transmitted without being interleaved, the TPC data may be accessed by slot unit in the receiving system.

**[00365]** Furthermore, the FIC data is transmitted through the FIC and is provided to enable a fast service acquisition of data receivers, and the FIC data includes cross layer information between the physical layer and the upper layer(s).

**[00366]** FIG. 30 illustrates a detailed block diagram of the signaling encoder 304 according to the present invention.

**[00367]** Referring to FIG. 34, the signaling encoder 304 includes a TPC encoder 561, an FIC encoder 562, a block interleaver 563, a multiplexer 564, a signaling randomizer 565, and an iterative turbo encoder 566.

**[00368]** The TPC encoder 561 receives 10-bytes of TPC data and performs (18,10)-RS encoding on the 10-bytes of TPC data, thereby adding 8 bytes of RS parity data to the 10 bytes of TPC data. The 18 bytes of RS-encoded TPC data are outputted to the multiplexer 564.

**[00369]** The FIC encoder 562 receives 37-bytes of FIC data and performs (51,37)-RS encoding on the 37-bytes of FIC data, thereby adding 14 bytes of RS parity data to the 37 bytes of FIC data. Thereafter, the 51 bytes of RS-encoded FIC data are inputted to the block interleaver 563, thereby being interleaved in predetermined block units. Herein, the block interleaver 563 corresponds to a variable length block interleaver. The block interleaver 563 interleaves the FIC data within each sub-frame in  $TNoG(\text{column}) \times 51(\text{row})$  block units and then outputs the interleaved data to the multiplexer 564. Herein, the TNoG corresponds to the total number of data groups being assigned to a sub-frame. The block interleaver 563 is synchronized with the first set of FIC data in each sub-frame.

**[00370]** The block interleaver 563 writes 51 bytes of incoming (or inputted) RS codewords in a row direction (i.e., row-by-row) and left-to-right and up-to-down directions and reads 51 bytes of RS codewords in a column direction (i.e., column-by-column) and left-to-right and up-to-down directions, thereby outputting the RS codewords.

**[00371]** The multiplexer 564 multiplexes the RS-encoded TPC data from the TPC encoder 561 and the block-interleaved FIC data from the block interleaver 563 along a time axis. Then, the multiplexer 564 outputs 69 bytes of the multiplexed data to the signaling randomizer 565.

**[00372]** The signaling randomizer 565 randomizes the multiplexed data and outputs the randomized data to the iterative turbo encoder 566. The signaling randomizer 565 may use the same generator polynomial of the randomizer used for mobile service data. Also, initialization occurs in each data group.

**[00373]** The iterative turbo encoder 566 corresponds to an inner encoder performing iterative turbo encoding in a PCCC method on the randomized data (i.e., signaling information data). The iterative turbo encoder 566 may include 6 even component encoders and 6 odd component encoders.

**[00374]** FIG. 31 illustrates an example of a syntax structure of TPC data being inputted to the TPC encoder 561.

**[00375]** The TPC data are inserted in the signaling information area of each data group and then transmitted. The TPC data may include a sub-frame\_number field, a slot\_number field, a parade\_id field, a starting\_group\_number (SGN) field, a number\_of\_groups (NoG) field, a parade\_repetition\_cycle (PRC) field, an RS\_frame\_mode field, an RS\_code\_mode\_primary field, an RS\_code\_mode\_secondary field, an SCCC\_block\_mode field, an SCCC\_outer\_code\_mode\_A field, an SCCC\_outer\_code\_mode\_B field, an SCCC\_outer\_code\_mode\_C field,

an `SCCC_outer_code_mode_D` field, an `FIC_version` field, a `parade_continuity_counter` field, and a `TNoG` field.

**[00376]** The `Sub-Frame_number` field corresponds to the current Sub-Frame number within the M/H frame, which is transmitted for M/H frame synchronization. The value of the `Sub-Frame_number` field may range from 0 to 4.

**[00377]** The `Slot_number` field indicates the current slot number within the sub-frame, which is transmitted for M/H frame synchronization. Also, the value of the `Sub-Frame_number` field may range from 0 to 15.

**[00378]** The `Parade_id` field identifies the parade to which this group belongs. The value of this field may be any 7-bit value. Each parade in a M/H transmission shall have a unique `Parade_id` field. Communication of the `Parade_id` between the physical layer and the management layer may be performed by means of an `Ensemble_id` field formed by adding one bit to the left of the `Parade_id` field. If the `Ensemble_id` field is used for the primary Ensemble delivered through this parade, the added MSB shall be equal to '0'. Otherwise, if the `Ensemble_id` field is used for the secondary ensemble, the added MSB shall be equal to '1'. Assignment of the `Parade_id` field values may occur at a convenient level of the system, usually in the management layer.

**[00379]** The `starting_group_number` (SGN) field shall be the first `Slot_number` for a parade to which this group belongs, as determined by Equation 1 (*i.e.*, after the Slot numbers for all preceding parades have been calculated). The SGN and NoG shall be used according to Equation 1 to obtain the slot numbers to be allocated to a parade within the sub-frame.

**[00380]** The `number_of_Groups` (NoG) field shall be the number of groups in a sub-frame assigned to the parade to which this group belongs, minus 1, *e.g.*, NoG = 0 implies that one group is allocated (or assigned) to this parade in a sub-frame. The value of NoG may range from 0 to 7. This limits the amount of

data that a parade may take from the main (legacy) service data, and consequently the maximum data that can be carried by one parade. The slot numbers assigned to the corresponding Parade can be calculated from SGN and NoG, using Equation 1. By taking each parade in sequence, the specific slots for each parade will be determined, and consequently the SGN for each succeeding parade. For example, if for a specific parade SGN = 3 and NoG = 3 (010b for 3-bit field of NoG), substituting  $i = 3, 4, \text{ and } 5$  in Equation 1 provides slot numbers 12, 2, and 6.

**[00381]** The Parade\_repetition\_cycle (PRC) field corresponds to the cycle time over which the parade is transmitted, minus 1, specified in units of M/H frames, as described in Table 12.

**[00382]** Table 12

| PRC | Description                                               |
|-----|-----------------------------------------------------------|
| 000 | This parade shall be transmitted once every M/H frame.    |
| 001 | This parade shall be transmitted once every 2 M/H frames. |
| 010 | This parade shall be transmitted once every 3 M/H frames. |
| 011 | This parade shall be transmitted once every 4 M/H frames. |
| 100 | This parade shall be transmitted once every 5 M/H frames. |
| 101 | This parade shall be transmitted once every 6 M/H frames. |
| 110 | This parade shall be transmitted once every 7 M/H frames. |
| 111 | Reserved                                                  |

**[00383]** For example, if PRC field value is equal to '001', this indicates that the parade shall be transmitted once every 2 M/H frame.

- [00384]** The RS\_Frame\_mode field shall be as defined in Table 1. The RS\_Frame\_mode field represents that one parade transmits one RS frame or two RS frames.
- [00385]** The RS\_code\_mode\_primary field shall be the RS code mode for the primary RS frame. Herein, the RS\_code\_mode\_primary field is defined in Table 6.
- [00386]** The RS\_code\_mode\_secondary field shall be the RS code mode for the secondary RS frame. Herein, the RS\_code\_mode\_secondary field is defined in Table 6.
- [00387]** The SCCC\_Block\_mode field represents how M/H blocks within a data group are assigned to SCCC block. The SCCC\_Block\_mode field shall be as defined in Table 7.
- [00388]** The SCCC\_outer\_code\_mode\_A field corresponds to the SCCC outer code mode for Region A within a data group. The SCCC outer code mode is defined in Table 8.
- [00389]** The SCCC\_outer\_code\_mode\_B field corresponds to the SCCC outer code mode for Region B within the data group.
- [00390]** The SCCC\_outer\_code\_mode\_C field corresponds be the SCCC outer code mode for Region C within the data group.
- [00391]** And, the SCCC\_outer\_code\_mode\_D field corresponds to the SCCC outer code mode for Region D within the data group.
- [00392]** The FIC\_version field represents a version of FIC data.
- [00393]** The Parade\_continuity\_counter field counter may increase from 0 to 15 and then repeat its cycle. This counter shall increment by 1 every (PRC+1) M/H frames. For example, as shown in Table 12, PRC = 011 (decimal 3) implies that Parade\_continuity\_counter increases every fourth M/H frame.
- [00394]** The TNoG field may be identical for all sub-frames in an M/H Frame.
- [00395]** However, the information included in the TPC data presented herein is merely exemplary. And, since the adding or deleting of information included in the TPC may be easily adjusted and modified by one skilled in the art, the present

invention will, therefore, not be limited to the examples set forth herein.

**[00396]** Since the TPC data (excluding the Sub-Frame\_number field and the Slot\_number field) for each parade do not change their values during an M/H frame, the same information is repeatedly transmitted through all M/H groups belonging to the corresponding parade during an M/H frame. This allows very robust and reliable reception of the TPC data. Because the Sub-Frame\_number and the Slot\_number are increasing counter values, they also are robust due to the transmission of regularly expected values. Furthermore, the FIC data is provided to enable a fast service acquisition of data receivers, and the FIC information includes cross layer information between the physical layer and the upper layer(s).

**[00397]** FIG. 32 illustrates an example of a transmission scenario of the TPC data and the FIC data. The values of the Sub-Frame\_number field, Slot\_number field, Parade\_id field, Parade\_repetition\_cycle field, and Parade\_continuity\_counter field may corresponds to the current M/H frame throughout the 5 sub-frames within a specific M/H frame. Some of TPC parameters and FIC data are signaled in advance.

**[00398]** The SGN, NoG and all FEC modes may have values corresponding to the current M/H frame in the first two sub-frames. The SGN, NoG and all FEC modes may have values corresponding to the frame in which the parade next appears throughout the 3<sup>rd</sup>, 4<sup>th</sup> and 5<sup>th</sup> sub-frames of the current M/H frame. This enables the M/H receivers to receive (or acquire) the transmission parameters in advance very reliably.

**[00399]** For example, when Parade\_repetition\_cycle = '000', the values of the 3<sup>rd</sup>, 4<sup>th</sup>, and 5<sup>th</sup> sub-frames of the current M/H frame correspond to the next M/H frame. Also, when Parade\_repetition\_cycle = '011', the values of the 3<sup>rd</sup>, 4<sup>th</sup>, and 5<sup>th</sup> sub-frames of the current M/H frame correspond to the 4<sup>th</sup> M/H frame and beyond.

- [00400]** The FIC\_version field and the FIC\_data field may have values that apply to the current M/H Frame during the 1<sup>st</sup> sub-frame and the 2<sup>nd</sup> sub-frame, and they shall have values corresponding to the M/H frame immediately following the current M/H frame during the 3<sup>rd</sup>, 4<sup>th</sup>, and 5<sup>th</sup> sub-frames of the current M/H frame.
- [00401]** Meanwhile, FIC data being transmitted through the FIC, i.e., an FIC chunk uses its fast characteristic so as to deliver mapping (or binding) information between a mobile service and an ensemble to the receiving system. More specifically, the FIC chunk corresponds to signaling data used for enabling the receiving system to swiftly find an ensemble that delivers a wanted (or desired) mobile service and to swiftly receive RS frames of the corresponding ensemble. At this point, the FIC chunk is segmented into a plurality of FIC segment payloads and a plurality of FIC segments are formed by added each FIC segment header to each FIC segment payload. Furthermore, one FIC segment is transmitted through one data group.
- [00402]** FIG. 33 illustrates a syntax structure of an FIC chunk that maps the relation between a mobile service and an ensemble through the FIC.
- [00403]** Herein, the FIC chunk consists of a 5-byte FIC chunk header and an FIC chunk payload having variable-length.
- [00404]** FIG. 34 illustrates a syntax structure of an FIC chunk header according to an embodiment of the present invention.
- [00405]** Herein, the FIC chunk header signals a non-backward compatible major protocol version change in a corresponding FIC chunk and also signals a backward compatible minor protocol version change. Furthermore, the FIC chunk header also signals the length for an extension of an FIC chunk header, the length for an extension of an ensemble loop header,

and the length for an extension of a mobile service loop that can be generated by a minor protocol version change.

**[00406]** According to an embodiment of the present invention, a receiver (or receiving system) that can adopt the corresponding minor protocol version change may process the corresponding extension field, whereas a legacy (or conventional) receiver that cannot adopt the corresponding minor protocol version change may skip the corresponding extension field by using each of the corresponding length information. For example, in case of a receiving system that can accept the corresponding minor protocol version change, the directions given in the corresponding extension field may be known. Furthermore, the receiving system may perform operations in accordance with the directions given in the corresponding extension field.

**[00407]** According to an embodiment of the present invention, a minor protocol version change in the FIC chunk is performed by inserting additional fields at the respective end portion of the FIC chunk header, the ensemble loop header, and the mobile service loop included in the previous minor protocol version FIC chunk. According to an embodiment of the present invention, in any other case, or when the length of the additional fields cannot be expressed (or indicated) by each extension length within the FIC chunk header, or when a specific field within the FIC chunk payload is missing (or cannot be found), or when the number of bits being assigned to the corresponding field or the definition of the corresponding field is changed (or altered), the major protocol version of the corresponding FIC chunk is updated.

**[00408]** Also, the FIC chunk header signals whether the data of a corresponding FIC chunk payload carry mapping information between an ensemble and a mobile service within the current M/H frame, or whether the data of a corresponding FIC chunk payload carry mapping information between an ensemble and a

mobile service within the next M/H frame. Furthermore, the FIC chunk header also signals the number of transport stream IDs of a mobile service through which the current FIC chunk is being transmitted and the number of ensembles being transmitted through the corresponding mobile service.

**[00409]** Accordingly, for this, the FIC chunk header may include an FIC\_major\_protocol\_version field, an FIC\_minor\_protocol\_version field, an FIC\_chunk\_header\_extension\_length field, an ensemble\_loop\_header\_extension\_length field, an M/H\_service\_loop\_extension\_length field, a current\_next\_indicator field, a transport\_stream\_id field, and a num\_ensembles field.

**[00410]** The FIC\_major\_protocol\_version field corresponds to a 2-bit unsigned integer field that represents the major version level of an FIC chunk syntax. A change in the major version level shall indicate a change in a non-backward-compatible level. When the FIC\_major\_protocol\_version field is updated, legacy (or conventional) receivers, which can process the prior major protocol version of an FIC chunk protocol, shall avoid processing the FIC chunk.

**[00411]** The FIC\_minor\_protocol\_version field corresponds to a 3-bit unsigned integer field that represents the minor version level of an FIC chunk syntax. When it is assumed that the major version level remains the same, a change in the minor version level shall indicate a change in a backward-compatible level. More specifically, when the FIC\_minor\_protocol\_version field is updated, legacy (or conventional) receivers, which can process the same major version of the FIC chunk protocol, may process a portion of the FIC chunk.

**[00412]** The FIC\_Chunk\_header\_extension\_length field corresponds to a 3-bit unsigned integer field identifying the length of FIC chunk header extension bytes, which are

generated by the minor protocol version update of the corresponding FIC chunk. Herein, the extension bytes are appended (or added) at the end of the corresponding FIC chunk header.

**[00413]** The `ensemble_header_extension_length` field corresponds to a 3-bit unsigned integer field identifying the length of the ensemble header extension bytes, which are generated by the minor protocol version update of the corresponding FIC chunk. Herein, the extension bytes are appended (or added) at the end of the corresponding ensemble loop header.

**[00414]** Also, the `M/H_service_loop_extension_length` field corresponds to a 4-bit unsigned integer field identifying the length of the ensemble header extension bytes, which are generated by the minor protocol version update of the M/H service loop. Herein, the extension bytes are appended (or added) at the end of the corresponding M/H service loop.

**[00415]** For example, it is assumed that the FIC chunk includes 2 ensembles (*i.e.*, ensemble 0 and ensemble 1). More specifically, it is assumed that two mobile services are transmitted through ensemble 0, and one mobile service is transmitted through ensemble 1. At this point, when the minor protocol version of the FIC chunk is changed, and the FIC chunk header is expanded by 1 byte, the `FIC_chunk_header_extension_length` field is marked as '001'. In this case, a 1-byte expansion field (*i.e.*, `FIC_Chunk_header_extension_bytes` field) is added at the end of the FIC chunk header. Also, the legacy receiver skips the 1-byte expansion field, which is added at the end of the FIC chunk header, without processing the corresponding expansion field.

**[00416]** Additionally, when the ensemble loop header within the FIC chunk is expanded by 2 bytes, the `ensemble_loop_header_extension_length` field is marked as '010'.

In this case, a 2-byte expansion field (*i.e.*, Ensemble\_loop\_header\_extension\_bytes field) is respectively added at the end of the ensemble 0 loop header and at the end of the ensemble 1 loop header. Also, the legacy receiver skips the 2-byte expansion fields, which are respectively added at the end of the ensemble 0 loop header and at the end of the ensemble 1 loop header, without processing the corresponding 2-byte expansion fields.

**[00417]** Furthermore, when the mobile service loop of the FIC chunk is expanded by 1 byte, the M/H\_service\_loop\_extension\_length field is marked as '001'. In this case, a 1-byte expansion field (*i.e.*, M/H\_service\_loop\_extension\_bytes field) is respectively added at the end of 2 mobile service loops being transmitted through ensemble 0 loop and at the end of 1 mobile service loop being transmitted through the ensemble 1 loop. And, the legacy receiver skips the 1-byte expansion fields, which are respectively added at the end of 2 mobile service loops being transmitted through ensemble 0 loop and at the end of 1 mobile service loop being transmitted through the ensemble 1 loop, without processing the corresponding 1-byte expansion fields.

**[00418]** As described above, when the FIC\_minor\_protocol version field is changed, a legacy (or conventional) receiver (*i.e.*, a receiver that cannot adopt the minor protocol version change in the corresponding FIC chunk) processes the fields apart from the extension field. Thereafter, the legacy receiver uses the FIC\_chunk\_header\_extension\_length field, the ensemble\_loop\_header\_extension\_length field, and the M/H\_service\_loop\_extension\_length field, so as to skip the corresponding expansion fields without processing the corresponding fields. When using a receiving system that can adopt the corresponding minor protocol version change of the FIC chunk, each length field is used to process even the corresponding expansion field.

**[00419]** The `current_next_indicator` field corresponds to a 1-bit indicator, which, when set to '1', indicates that the corresponding FIC chunk is currently applicable. Alternatively, when the `current_next_indicator` field is set to '0', the `current_next_indicator` field indicates that the corresponding FIC chunk will be applicable for the next M/H frame. Herein, when the `current_next_indicator` field is set to '0', the most recent version of the FIC chunk being transmitted with the `current_next_indicator` field set to '1' shall be currently applicable. More specifically, when the `current_next_indicator` field value is set to '1', this indicates that the corresponding FIC chunk transmits the signaling data of the current M/H frame. Further, when the `current_next_indicator` field value is set to '0', this indicates that the corresponding FIC chunk transmits the signaling data of the next M/H frame. When reconfiguration occurs, wherein the mapping information between the ensemble within the current M/H frame and the mobile service differs from the ensemble within the next M/H frame and the mobile service, the M/H frame prior to reconfiguration is referred to as the current M/H frame, and the M/H frame following reconfiguration is referred to as the next M/H frame.

**[00420]** The `transport_stream_id` field corresponds to a 16-bit unsigned integer number field, which serves as a label for identifying the corresponding M/H broadcast. The value of the corresponding `transport_stream_id` field shall be equal to the value of the `transport_stream_id` field included in the program association table (PAT) within the MPEG-2 transport stream of a main ATSC broadcast.

**[00421]** The `num_ensembles` field corresponds to an 8-bit unsigned integer field, which indicates the number of M/H ensembles carried through the corresponding physical transmission channel.

**[00422]** FIG. 35 illustrates an exemplary syntax structure of an FIC chunk payload according to an embodiment of the present invention.

**[00423]** For each ensemble corresponding to the num\_ensembles field value within the FIC chunk header of FIG. 34, the FIC chunk payload includes configuration information of each ensemble and information on mobile services being transmitted through each ensemble.

**[00424]** The FIC chunk payload consists of an ensemble loop and a mobile service loop below the ensemble loop. The FIC chunk payload enables the receiver to determine through which ensemble a requested (or desired) mobile service is being transmitted. (This process is performed via mapping between the ensemble\_id field and the M/H\_service\_id field.) Thus, the receiver may receive RS frames belonging to the corresponding ensemble.

**[00425]** In order to do so, the ensemble loop of the FIC chunk payload may include an ensemble\_id field, an ensemble\_protocol\_version field, an SLT\_ensemble\_indicator field, a GAT\_ensemble\_indicator field, an MH\_service\_signaling\_channel\_version field, and a num\_M/H\_services field, which are collectively repeated as many times as the num\_ensembles field value. The mobile service loop may include an MH\_service\_id field, a multi\_ensemble\_service field, an MH\_service\_status field, and an SP\_indicator field, which are collectively repeated as many times as the num\_M/H\_services field.

**[00426]** The ensemble\_id field corresponds to an 8-bit unsigned integer field, which indicates a unique identifier of the corresponding ensemble. For example, the ensemble\_id field may be assigned with values within the range '0x00' to '0x7F'. The ensemble\_id field group (or associate) the mobile services with the respective ensemble. Herein, it is preferable that the value of the ensemble\_id field is derived from the

parade\_id field carried (or transmitted) through the TPC data. If the corresponding ensemble is transmitted through a primary RS frame, the most significant bit is set to '0', and the remaining least significant bits are used as the parade\_id field value of the corresponding parade. Meanwhile, if the corresponding ensemble is transmitted through a secondary RS frame, the most significant bit is set to '0', and the remaining least significant bits are used as the parade\_id field value of the corresponding parade.

**[00427]** The ensemble\_protocol\_version field corresponds to a 5-bit field, which specifies a version of the corresponding ensemble structure.

**[00428]** The SLT\_ensemble\_indicator field is a 1-bit field, which indicates whether or not the SLT is being transmitted to the service signaling channel of the corresponding ensemble. For example, when the SLT\_ensemble\_indicator field value is equal to '1', this may indicate that the SLT is being transmitted to the service signaling channel. On the other hand, when the SLT\_ensemble\_indicator field value is equal to '0', this may indicate that the SLT is not being transmitted.

**[00429]** The GAT\_ensemble\_indicator field is also a 1-bit field, which indicates whether or not the GAT is being transmitted to the service signaling channel of the corresponding ensemble. For example, when the GAT\_ensemble\_indicator field value is equal to '1', this may indicate that the GAT is being transmitted to the service signaling channel. On the other hand, when the GAT\_ensemble\_indicator field value is equal to '0', this may indicate that the GAT is not being transmitted.

**[00430]** The MH\_service\_signaling\_channel\_version field corresponds to a 5-bit field, which indicates a version number of the service signaling channel of the corresponding ensemble.

**[00431]** The num\_M/H\_services field corresponds to an 8-bit unsigned integer field, which represents the number of mobile

(i.e., M/H) services carried through the corresponding M/H ensemble.

**[00432]** For example, when the minor protocol version within the FIC chunk header is changed, and when an extension field is added to the ensemble loop header, the corresponding extension field is added immediately after the num\_M/H\_services field. According to another embodiment of the present invention, if the num\_M/H\_services field is included in the mobile service loop, the corresponding extension field that is to be added in the ensemble loop header is added immediately after the M/H\_service\_configuration\_version field.

**[00433]** The M/H\_service\_id field of the mobile service loop corresponds to a 16-bit unsigned integer number, which identifies the corresponding M/H service. The value (or number) of the M/H\_service\_id field shall be unique within the mobile (M/H) broadcast.

**[00434]** The multi\_ensemble\_service field is a 2-bit enumerated field, which indicates whether the corresponding mobile (M/H) service is transmitted through (or over) one ensemble, or whether the corresponding mobile (M/H) service is transmitted through (or over) multiple ensembles. Also, the value of the multi\_ensemble\_service field indicates whether or not the mobile service is valid (or rendered meaningfully) only for the mobile service portion being transmitted through (or over) the corresponding ensemble.

**[00435]** The M/H\_service\_status field corresponds to a 2-bit enumerated field, which identifies the status of the corresponding M/H service. For example, the most significant bit of the M/H\_service\_status field indicates whether the corresponding M/H service is active (when set to '1') or inactive (when set to '0'). Furthermore, the least significant bit indicates whether the corresponding M/H service is hidden (when set to '1') or not (when set to '0').

[00436] The SP\_indicator field corresponds to a 1-bit field, which, when set to '1', indicates whether or not service protection is applied to at least one of the components required for providing a significant presentation of the corresponding M/H service.

[00437] For example, when the minor protocol version of the FIC chunk is change, and if an expansion field is added to the mobile service loop, the expansion field is added after the SP\_indicator field.

[00438] Also, the FIC chunk payload may include an FIC\_chunk\_stuffing() field. Stuffing of the FIC\_chunk\_stuffing() field may exist in an FIC-Chunk, to keep the boundary of the FIC-Chunk to be aligned with the boundary of the last FIC-Segment among FIC segments belonging to the FIC chunk. The length of the stuffing is determined by how much space is left after parsing through the entire FIC-Chunk payload preceding the stuffing.

[00439] At this point, the transmitting system (not shown) according to the present invention divides the FIC chunk into multiple FIC segment payloads and forms a plurality of FIC segments by adding each FIC segment header to each FIC segment payload, thereby outputting the FIC segments to the receiving system in FIC segment units. The size of each FIC segment unit is 37 bytes, and each FIC segment consists of a 2-byte FIC segment header and a 35-byte FIC segment payload. More specifically, an FIC chunk, which is configured of an FIC chunk header and an FIC chunk payload, is segmented by units of 35 bytes. Also, an FIC segment is configured by adding a 2-byte FIC segment header in front of each segmented 35-byte unit.

[00440] According to an embodiment of the present invention, the length of the FIC chunk payload is variable. Herein, the length of the FIC chunk varies depending upon the number of ensembles being transmitted through the corresponding physical

transmission channel and the number of mobile services included in each ensemble.

**[00441]** Also, the FIC chunk payload may include stuffing data. In this case, the stuffing data are used for the boundary alignment of the FIC chunk and the last FIC-Segment, among FIC segments belonging to the FIC chunk, according to the embodiment of the present invention. Accordingly, by minimizing the length of the stuffing data, unnecessary wasting of FIC segments can be reduced.

**[00442]** At this point, the number of stuffing data bytes being inserted in the FIC chunk can be calculated by using Equation 6 below.

**[00443]** Equation 6

The number of stuffing data bytes =  $35 - j$

$j = (5 + \text{the number of signaling data bytes being inserted in the FIC chunk payload}) \bmod 35$

**[00444]** For example, when the added total length of the 5-byte header within the FIC chunk and signaling data, which is to be inserted in the payload within the FIC chunk, is equal to 205 bytes, the payload of the FIC chunk may include 5 bytes of stuffing data because  $j$  is equal to 30 in Equation 6. Also, the length of the FIC chunk payload including the stuffing data is equal to 210 bytes. Thereafter, the FIC chunk is divided into 6 FIC segments, which are then transmitted. At this point, a segment number is sequentially assigned to each of the 6 FIC segments divided from the FIC chunk.

**[00445]** Furthermore, the present invention may transmit the FIC segments divided from a single FIC chunk to a single sub-frame, or may transmit the divided FIC segments to multiple sub-frames. If the FIC chunk is divided and transmitted to multiple sub-frames, signaling data, which are required even when the amount of data that are to be transmitted through the FIC chunk is larger than the amount of FIC segments being transmitted through a single sub-frame (this case corresponds

to when multiple services having very low bit rates are being executed), may all be transmitted through the FIC chunk.

**[00446]** Herein, the FIC segment numbers represent FIC segment numbers within each FIC chunk, and not the FIC segment number within each sub-frame. Thus, the subordinate relation between the FIC chunk and the sub-frame can be eliminated, thereby reducing excessive waste of FIC segments.

**[00447]** Furthermore, the present invention may add a null FIC segment. Despite the repeated transmission of the FIC chunk, and when stuffing is required in the corresponding M/H frame, the null FIC segment is used for the purpose of processing the remaining FIC segments. For example, it is assumed that TNoG is equal to '3' and that the FIC chunk is divided into 2 FIC segments. Herein, when the FIC chunk is repeatedly transmitted through 5 sub-frames within a single M/H frame, only 2 FIC segments are transmitted through one of the 5 sub-frames (e.g., the sub-frame chronologically placed in the last order). In this case, one null FIC segment is assigned to the corresponding sub-frame, thereby being transmitted. More specifically, the null FIC segment is used for aligning the boundary of the FIC chunk and the boundary of the M/H frame. At this point, since the null FIC segment is not an FIC segment divided from the FIC chunk, an FIC segment number is not assigned to the null FIC segment.

**[00448]** In the present invention, when a single FIC chunk is divided into a plurality of FIC segments, and when the divided FIC segments are included in each data group of at least one sub-frame within the M/H frame, so as to be transmitted, the corresponding FIC segments are allocated in a reversed order starting from the last sub-frame within the corresponding M/H frame. According to an embodiment of the present invention, in case a null FIC segment exists, the null FIC segment is positioned in the sub-frame within the M/H frame, so that the

corresponding null FIC segment can be transmitted as the last (or final) segment.

**[00449]** At this point, in order to enable the receiving system to discard the null FIC segment without having to process the corresponding null FIC segment, identification information that can identify (or distinguish) the null FIC segment is required.

**[00450]** According to an embodiment of the present invention, the present invention uses the FIC\_segment\_type field within the header of the null FIC segment as the identification information for identifying the null FIC segment. In this embodiment, the value of the FIC\_segment\_type field within the null FIC segment header is set to '11', so as to identify the corresponding null FIC segment. More specifically, when the FIC\_segment\_type field value within the null FIC segment header is set to '11' and transmitted to the receiving system, the receiving system may discard the payload of the FIC segment having the FIC\_segment\_type field value set to '11' without having to process the corresponding FIC segment payload. Herein, the value '11' is merely an exemplary value given to facilitate and simplify the understanding of the present invention. As long as a pre-arrangement between the receiving system and the transmitting system is established, any value that can identify the null FIC segment may be given to the FIC\_segment\_type field. Therefore, the present invention will not be limited only to the example set presented herein. Furthermore, the identification information that can identify the null FIC segment may also be indicated by using another field within the FIC segment header.

**[00451]** FIG. 36 illustrates an exemplary syntax structure of an FIC segment header according to an embodiment of the present invention.

**[00452]** Herein, the FIC segment header may include an FIC\_segment\_type field, an FIC\_chunk\_major\_protocol\_version

field, a `current_next_indicator` field, an `error_indicator` field, an `FIC_segment_num` field, and an `FIC_last_segment_num` field. Each field will now be described as follows.

**[00453]** The `FIC_segment_type` field corresponds to a 2-bit field, which, when set to `'00'` indicates that the corresponding FIC segment is carrying a portion of an FIC chunk. Alternatively, when the `FIC_segment_type` field is set to `'11'`, the `FIC_segment_type` field indicates that the corresponding FIC segment is a null FIC segment, which transmits stuffing data. Herein, the remaining values are reserved for future use.

**[00454]** The `FIC_Chunk_major_protocol_version` field corresponds to a 2-bit field, which indicates a major protocol version of the corresponding FIC chunk. At this point, the value of the `FIC_Chunk_major_protocol_version` field should be the same as the value of the `FIC_major_protocol_version` field within the corresponding FIC chunk header. Since reference may be made to the description of the FIC chunk header shown in FIG. 34, a detailed description of the major protocol version of the FIC chunk syntax will be omitted for simplicity.

**[00455]** The `current_next_indicator` field corresponds to a 1-bit indicator, which, when set to `'1'`, shall indicate that the corresponding FIC segment is carrying a portion of the FIC chunk, which is applicable to the current M/H frame. Alternatively, when the value of the `current_next_indicator` field is set to `'0'`, the `current_next_indicator` field shall indicate that the corresponding FIC segment is carrying a portion of the FIC chunk, which will be applicable for the next M/H frame.

**[00456]** The `error_indicator` field corresponds to a 1-bit field, which indicates whether or not an error has occurred in the corresponding FIC segment during transmission. Herein, the `error_indicator` field is set to `'1'`, when an error has occurred. And, the `error_indicator` field is set to `'0'`, when

an error does not exist (or has not occurred). More specifically, during the process of configuring the FIC segment, when a non-recovered error exists, the error\_indicator field is set to '1'. More specifically, the error\_indicator field enables the receiving system to recognize the existence (or presence) of an error within the corresponding FIC segment.

**[00457]** The FIC\_segment\_num field corresponds to a 4-bit unsigned integer number field, which indicates a number of the corresponding FIC segment. For example, if the corresponding FIC segment is the first FIC segment of the FIC chunk, the value of the FIC\_segment\_num field shall be set to '0x0'. Also, if the corresponding FIC segment is the second FIC segment of the FIC chunk, the value of the FIC\_segment\_num field shall be set to '0x1'. More specifically, the FIC\_segment\_num field shall be incremented by one with each additional FIC segment in the FIC chunk. Herein, if the FIC chunk is divided into 4 FIC segments, the FIC\_segment\_num field value of the last FIC segment within the FIC chunk will be indicated as '0x3'.

**[00458]** The FIC\_last\_segment\_num field corresponds to a 4-bit unsigned integer number field, which indicates the number of the last FIC segment (*i.e.*, the FIC segment having the highest FIC\_segment\_num field value) within a complete FIC chunk.

**[00459]** In the conventional method, FIC segment numbers are sequentially assigned (or allocated) for each FIC segment within one sub-frame. Therefore, in this case, the last FIC segment number always matches with the TNoG (*i.e.*, the last FIC segment number is always equal to the TNoG). However, when using the FIC number assignment method according to the present invention, the last FIC segment number may not always match with the TNoG. More specifically, the last FIC segment number may match with the TNoG, or the last FIC segment number may not match with the TNoG. The TNoG represents a total

number of data groups that are allocated (or assigned) to a single sub-frame. For example, when the TNoG is equal to '6', and when the FIC chunk is divided into 8 FIC segments, the TNoG is equal to '6', and the last FIC segment number is '8'.

**[00460]** According to another embodiment of the present invention, the null FIC segment may be identified by using the value of the FIC\_segment\_num field within the FIC segment header. More specifically, since an FIC segment number is not assigned to the null FIC segment, the transmitting system allocates null data to the FIC\_segment\_num field value of the null FIC segment, and the receiving system may allow the FIC segment having null data assigned to the FIC\_segment\_num field value to be recognized as the null FIC segment. Herein, instead of the null data, data pre-arranged by the receiving system and the transmitting system may be assigned to the FIC\_segment\_num field value, instead of the null data.

**[00461]** As described above, the FIC chunk is divided into a plurality of FIC segments, thereby being transmitted through a single sub-frame or being transmitted through multiple sub-frames. Also, FIC segments divided from a single FIC chunk may be transmitted through a single sub-frame, or FIC segments divided from multiple single FIC chunks may be transmitted through a single sub-frame. At this point, the number assigned to each FIC segment corresponds to a number within the corresponding FIC chunk (i.e., the FIC\_seg\_number value), and not the number within the corresponding sub-frame. Also, the null FIC segment may be transmitted for aligning the boundary of the M/H frame and the boundary of the FIC chunk. At this point, an FIC segment number is not assigned to the null FIC segment.

**[00462]** As described above, one FIC chunk may be transmitted through multiple sub-frames, or multiple FIC chunks may be transmitted through a single sub-frame. However, according to

the embodiment of the present invention, the FIC segments are interleaved and transmitted in sub-frame units.

**[00463]** Meanwhile, FIG. 37 illustrates an exemplary structure of a bit stream syntax of an SMT section which is included in the RS frame and then transmitted. Herein, the SMT section is configured in an MPEG-2 private section format for simplicity. However, the SMT section data may be configured in any possible format.

**[00464]** The SMT may provide access information of mobile services within an ensemble including the SMT. Also, the SMT may provide information required for the rendering of mobile services. Furthermore, the SMT may include at least one or more descriptors. Herein, other additional (or supplementary) information may be described by the descriptor.

**[00465]** At this point, the service signaling channel that transmits the SMT may further include another signaling table (e.g., GAT) in addition to the SMT.

**[00466]** Herein, according to the embodiment of the present invention, IP datagrams of the service signaling channel have the same well-known destination IP address and the same well-known destination UDP port number. Therefore, the SMT included in the service signaling data is distinguished (or identified) by a table identifier. More specifically, the table identifier may correspond to a `table_id` existing in the corresponding table or in a header of the corresponding table section. And, when required, the table identifier may further refer to a `table_id_extension` field, so as to perform the identification process. Exemplary fields that can be transmitted through the SMT section will now be described in detail.

**[00467]** A `table_id` field is an 8-bit table identifier, which may be set up as an identifier for identifying the SMT.

**[00468]** A `section_syntax_indicator` field corresponds to an indicator defining the section format of the SMT. For example, the `section_syntax_indicator` field shall be set to '0' to

always indicate that this table is derived from the "short" form of the MPEG-2 private section table format may correspond to MPEG long-form syntax.

**[00469]** A `private_indicator` field is a 1-bit field, which indicates whether or not the SMT follows (or is in accordance with) a private section.

**[00470]** A `section_length` field is a 12-bit field, which specifies the section length of the remaining SMT data bytes immediately following the `section_length` field.

**[00471]** A `table_id_extension` field corresponds to a table-dependent 16-bit field. Herein, the `table_id_extension` field corresponds to a logical portion of the `table_id` field providing the scope for the remaining fields. The `table_id_extension` field includes a `SMT_protocol_version` field and an `ensemble_id` field.

**[00472]** The `SMT_protocol_version` field corresponds to an 8-bit unsigned integer field. Herein, the `SMT_protocol_version` field indicates a protocol version for allowing the corresponding SMT to carry, in a future process, parameters that may be structure differently from those defined in the current protocol. Presently, the value of the `SMT_protocol_version` field shall be equal to zero(0). Non-zero values of the `SMT_protocol_version` field may be used by a future version of this standard to indicate structurally different tables.

**[00473]** The `ensemble_id` field corresponds to an 8-bit field. Herein, the ID values associated with the corresponding ensemble that can be assigned to the `ensemble_id` field may range from '0x00' and '0x3F'. It is preferable that the value of the `ensemble_id` field is derived from the TPC data of the `parade_id` field. When the corresponding ensemble is transmitted through a primary RS frame, the most significant bit (MSB) is set to '0', and the remaining 7 bits are used as the `parade_id` field value of the corresponding parade.

Meanwhile, when the corresponding ensemble is transmitted through a primary RS frame, the most significant bit (MSB) is set to '1', and the remaining 7 bits are used as the parade\_id field value of the corresponding parade.

**[00474]** A version\_number field corresponds to a 5-bit field, which specifies the version number of the SMT.

**[00475]** A current\_next\_indicator field corresponds to a 1-bit field indicating whether or not the SMT section is currently applicable.

**[00476]** A section\_number field is an 8-bit field specifying the number of the current SMT section.

**[00477]** A last\_section\_number field corresponds to an 8-bit field that specifies the number of the last section configuring the corresponding SMT.

**[00478]** And, a num\_MH\_services field corresponds to an 8-bit field, which specifies the number of mobile services in the corresponding SMT section.

**[00479]** Hereinafter, a number of 'for' loop (also referred to as mobile (M/H) service loop) statements equivalent to the number of mobile services corresponding to the num\_MH\_services field is performed so as to provide signaling information on multiple mobile services. More specifically, signaling information of the corresponding mobile service is indicated for each mobile service that is included in the SMT section. Herein, the following field information corresponding to each mobile service may be provided as described below.

**[00480]** An MH\_service\_id field corresponds to a 16-bit unsigned integer number, which can uniquely identify the corresponding mobile service within the scope of the corresponding SMT section.

**[00481]** A multi\_ensemble\_service field corresponds to a 2-bit field, which indicates whether the corresponding mobile service is transmitted through one or more ensembles. Since the multi\_ensemble\_service field has the same meaning as the

multi\_ensemble\_service field included in the FIC chunk, detailed description of the same will be omitted for simplicity.

**[00482]** An MH\_service\_status field corresponds to a 2-bit field, which can identify the status of the corresponding mobile service. Herein, the MSB indicates whether the corresponding mobile service is active ('1') or whether the corresponding mobile service is inactive ('0'). Also, the LSB indicates whether the corresponding mobile service is hidden ('1') or not hidden ('0').

**[00483]** An SP\_indicator field corresponds to a 1-bit field, which specifies service protection status of the corresponding mobile service. If the SP\_indicator field is set to '1', then service protection is applied to at least one of the components needed to provide a meaningful presentation of the corresponding service.

**[00484]** A short\_MH\_service\_name\_length field corresponds to a 3-bit field, which indicates the length of a short service name described in a short\_service\_name field in byte-length units.

**[00485]** The short\_MH\_service\_name field indicates the short name of the corresponding mobile service.

**[00486]** An MH\_service\_category field is a 6-bit field, which identifies the type category of the corresponding mobile service.

**[00487]** A num\_components field corresponds to a 5-bit field, which specifies the number of IP stream components in the corresponding mobile service.

**[00488]** An IP\_version\_flag field corresponds to a 1-bit indicator, which when set to '0' indicates that a source\_IP\_address field, an MH\_service\_destination\_IP\_address field, and a component\_destination\_IP\_address field correspond to IPv4 addresses. The value of '1' for the IP\_version\_flag field is reserved for any possible future indication that the

source\_IP\_address field, the MH\_service\_destination\_IP\_address field, and the component\_destination\_IP\_address field correspond to IPv6 addresses. However, the usage of IPv6 addressing is currently undefined.

**[00489]** A source\_IP\_address\_flag corresponds to a 1-bit Boolean flag, which indicates, when set, that a source IP address value for the corresponding service exists (or is present) so as to indicate a source specific multicast.

**[00490]** An MH\_service\_destination\_IP\_address\_flag corresponds to a 1-bit, which indicates, when set, that the corresponding IP stream component is transmitted through an IP datagram having a destination IP address different from that of the MH\_service\_destination\_IP\_address field. Therefore, when the MH\_service\_destination\_IP\_address\_flag is set, the receiving system may use the component\_destination\_IP\_address as the destination\_IP\_address in order to access the corresponding IP stream component. Furthermore, the receiving system ignores (or disregards) the MH\_service\_destination\_IP\_address field within the mobile service loop.

**[00491]** The source\_IP\_address field corresponds to a 32-bit field or a 128-bit field. When the source\_IP\_address\_flag is set to '1', the source\_IP\_address field is required to be interpreted (or analyzed). However, when the source\_IP\_address\_flag is set to '0', the source\_IP\_address field is not required to be interpreted (or analyzed). When the source\_IP\_address\_flag is set to '1', and when the IP\_version\_flag field is set to '0', the corresponding field indicates that the source\_IP\_address field indicates a 32-bit IPv4 address specifying the corresponding mobile service source. Alternatively, if the IP\_version\_flag field is set to '1', the source\_IP\_address field indicates a 32-bit IPv6 address specifying the corresponding mobile service source.

**[00492]** The MH\_service\_destination\_IP\_address field corresponds to a 32-bit field or a 128-bit field. When the MH\_service\_destination\_IP\_address\_flag field is set to '1', the MH\_service\_destination\_IP\_address\_flag is required to be interpreted (or analyzed). However, when the MH\_service\_destination\_IP\_address\_flag is set to '0', the MH\_service\_destination\_IP\_address\_flag is not required to be interpreted (or analyzed). Herein, if the MH\_service\_destination\_IP\_address\_flag is set to '1', and if the IP\_version\_flag field is set to '0', the MH\_service\_destination\_IP\_address field indicates a 32-bit destination IPv4 address for the corresponding mobile service. Alternatively, if the MH\_service\_destination\_IP\_address\_flag is set to '1', and if the IP\_version\_flag field is set to '1', the MH\_service\_destination\_IP\_address field indicates a 64-bit destination IPv6 address for the corresponding mobile service. In case the corresponding MH\_service\_destination\_IP\_address field cannot be interpreted, the component\_destination\_IP\_address field within a component loop shall be interpreted. And, in this case, the receiving system shall use the component\_destination\_IP\_address in order to access the IP stream component.

**[00493]** Meanwhile, the SMT according to the embodiment of the present invention provides information on multiple components using the 'for' loop statement.

**[00494]** Hereinafter, a number of 'for' loop (also referred to as component loop) statements equivalent to the number of components corresponding to the num\_component field value is performed so as to provide access information on multiple components. More specifically, access information of each component included in the corresponding mobile service is provided. In this case, the following field information on each component may be provided as described below.

**[00495]** A component\_source\_IP\_address\_flag field is a 1-bit field (or a 1-bit Boolean flag), which indicates, when set to '1', that the component\_source\_IP\_address field is present for this component.

**[00496]** More specifically, a mobile service may include diverse types of components, for example, a mobile service may include an audio component, or a mobile service may include a video component, or a mobile service may include a FLUTE component.

**[00497]** At this point, when the component\_source\_IP\_address\_flag field is set to '1', this signifies that the component\_source\_IP\_address field exists, and this field indicates a source IP address of an IP datagram carrying the corresponding component.

**[00498]** For example, when the component\_source\_IP\_address\_flag field of the FLUTE component is set to '1', the component\_source\_IP\_address field indicates the source IP address of the IP datagram carrying the FLUTE component.

**[00499]** According to an embodiment of the present invention, in case a service\_source\_IP\_address field within a mobile service loop and a component\_source\_IP\_address within a component loop both exist, yet if the field values are different from one another, the source IP address of the IP datagram of the corresponding component is acquired from the component\_source\_IP\_address field. More specifically, the service\_source\_IP\_address field within the mobile service loop is disregarded.

**[00500]** According to another embodiment of the present invention, in case the service\_source\_IP\_address field exists and the component\_source\_IP\_address field does not exist, the source IP address of the IP datagram of the corresponding component is acquired from the service\_source\_IP\_address field. And, in the opposite case, i.e., in case the

service\_source\_IP\_address field does not exist and the component\_source\_IP\_address field exists, the source IP address of the IP datagram of the corresponding component is acquired from the component\_source\_IP\_address field.

**[00501]** As described above, in the present invention, the component\_source\_IP\_address field exists when the component\_source\_IP\_address\_flag field value is equal to '1'. And, according to an embodiment of the present invention, when the component\_source\_IP\_address field exists, the source IP address of the IP datagram of the corresponding component is identical to the component\_source\_IP\_address field value. And, according to another embodiment of the present invention, in case the component\_source\_IP\_address field does not exist, the source IP address of the IP datagram of the corresponding component is identical to the service\_source\_IP\_address field value.

**[00502]** And, according to another embodiment of the present invention, in case the service\_source\_IP\_address field and the component\_source\_IP\_address field do not exist, the source IP address is not used when acquiring the IP datagram of the corresponding component.

**[00503]** An essential\_component\_indicator field is a 1-bit field, which indicates that the corresponding component is an essential component for the mobile service, when the essential\_component\_indicator field value is set to '1'. Otherwise, the essential\_component\_indicator field indicates that the corresponding component is an optional component. For example, in case of a basic layer audio stream and video stream, the essential\_component\_indicator field value is set to '1'. And, in case of the enhanced layer video stream, the essential\_component\_indicator field value is set to '0'.

**[00504]** A component\_destination\_IP\_address\_flag field corresponds to a 1-bit Boolean flag. When the component\_destination\_IP\_address\_flag field is set to '1',

this indicates that a component\_destination\_IP\_address exists for the corresponding component.

**[00505]** A port\_num\_count field corresponds to a 6-bit field, which indicates a UDP port number associated with the corresponding UDP/IP stream component. Herein, the destination UDP Port number value is increased by 1 starting from a destination\_UDP\_port\_num field value. The destination\_UDP\_port\_num field corresponds to a 16-bit field, which indicates a destination UDP port number for the corresponding IP stream component.

**[00506]** A component\_source\_IP\_address field corresponds to a 32-bit or 128-bit field, which exists when the value of the component\_source\_IP\_address\_flag field is equal to '1'. At this point, in case the IP\_version\_flag field is set to '0', the component\_source\_IP\_address field indicates a 32-bit source IPv4 address for the corresponding IP stream component. Also, in case the IP\_version\_flag field is set to '1', the component\_source\_IP\_address field indicates a 128-bit source IPv6 address for the corresponding IP stream component.

**[00507]** According to an embodiment of the present invention, in case the component\_source\_IP\_address field exists, the source IP address of an IP datagram of the corresponding component is acquired from the component\_source\_IP\_address field.

**[00508]** A component\_destination\_IP\_address field corresponds to a 32-bit field or a 128-bit field. When the IP\_version\_flag field is set to '0', the component\_destination\_IP\_address field indicates a 32-bit destination IPv4 address for the corresponding IP stream component.

**[00509]** Furthermore, when the IP\_version\_flag field is set to '1', the component\_destination\_IP\_address field indicates a 128-bit destination IPv6 address for the corresponding IP stream component. When this field is present, the destination address of the IP datagrams carrying the corresponding

component of the M/H service shall match the address in the component\_destination\_IP\_address field. Alternatively, when this field is not present, the destination address of the IP datagrams carrying the corresponding component shall match the address in the M/H\_service\_destination\_IP\_address field. The conditional use of the 128 bit-long address version of this field is to facilitate possible future usage of the IPv6, although the usage of the IPv6 is currently undefined.

**[00510]** A num\_component\_level\_descriptors field corresponds to a 4-bit field, indicating a number of descriptors providing additional information on the component level.

**[00511]** A number of component\_level\_descriptor() corresponding to the value of the num\_component\_level\_descriptors field is included in the component loop, so as to provide additional (or supplemental) information on the corresponding component.

**[00512]** A num\_MH\_service\_level\_descriptors field corresponds to a 4-bit field indicating a number of descriptors providing additional information of the corresponding mobile service level.

**[00513]** A number of service\_level\_descriptor() corresponding to the value of the num\_MH\_service\_level\_descriptors field is included in the mobile service loop, so as to provide additional (or supplemental) information on the mobile service.

**[00514]** A num\_ensemble\_level\_descriptors field corresponds to a 4-bit field, which indicates a number of descriptors providing additional information on ensemble levels.

**[00515]** Furthermore, a number of ensemble\_level\_descriptor() corresponding to the value of the num\_ensemble\_level\_descriptors field is included in the ensemble loop, so as to provide additional (or supplemental) information on the ensemble.

**[00516]** FIG. 38 illustrates an embodiment of a bit stream syntax structure of a component\_level\_descriptors(). The

`component_descriptor()` is used as one of the component level descriptor `component_level_descriptors()` of the NST and describes additional signaling information of the corresponding component.

**[00517]** The following is a description of each field of the `component_descriptor()`.

**[00518]** In FIG. 38, a `descriptor_tag` field (8-bit) is a descriptor identifier and it can be set as an identifier that identifies the `component_descriptor()`.

**[00519]** A `descriptor_length` field (8-bit) describes the remaining length of the descriptor starting after the `descriptor_length` field and to the end of this descriptor, in bytes.

**[00520]** A `component_type` field (7-bit) shall identify the encoding format of the component. The value may be any of the values assigned by IANA for the `payload_type` of an RTP/AVP stream, or it may be any of the values assigned by ATSC, or it may be a "dynamic value" in the range 96-127. For components consisting of media carried via RTP, the value of this field shall match the value in the `payload_type` field in the RTP header of the IP stream carrying this component. Note that additional values of the `component_type` field in the range of 43-71 can be defined in future versions of this standard.

**[00521]** A `component_encryption_flag` (1-bit) informs whether the corresponding component is encrypted or not.

**[00522]** A `Num_STKM_streams` field (8-bit) indicates the number STKM streams if `component_encryption_flag` has been encrypted. (The `num_STKM_streams` field (8-bit) is an 8-bit unsigned integer field that shall identify the number of STKM streams associated with this component.

**[00523]** A `STKM_stream_id` field (8-bit) is repeated as much as the field value of `Num_STKM_streams` and indicates a value that identifies a SKTM stream that can acquire a key required for decryption.

**[00524]** A component\_data (component\_type) element provides the encoding parameters and/or other parameters necessary for rendering this component. The structure of the component\_data is determined by the value of component\_type field.

**[00525]** For example, if the component\_type field value is 35 then component\_data (component\_type) field provides component data for H.264/AVC video stream.

**[00526]** In another example, if the component\_type field value is 38 then component\_data (component\_type) field provides data for FLUTE file delivery.

**[00527]** One mobile service can be included in multiple FLUTE sessions. Thus, one mobile service may be configured with plurality of FLUTE sessions. Each FLUTE session may be signaled using component\_data().

**[00528]**

**[00529]** Meanwhile, according to another embodiment of the present invention, a broadcast receiver to provide a user with a mobile broadcast service by processing a mobile broadcast signal provided over a mobile broadcast network, particularly the M/H broadcast network is allowed to provide disaster broadcasting (or disaster alert). To provide the disaster broadcast containing disaster information, signaling information for the disaster broadcast is necessary. Hereinafter, a description will be given of various examples of a method for signaling the signaling information for the disaster broadcast.

**[00530]** The signaling information may be signaled to the FIC chunk and/or the SMT. At this time, the signaling information signaled to the FIC chunk and the signaling information signaled to the SMT may be different from, or identical to each other, or parts thereof may duplicate.

**[00531]** In addition, the present invention defines a new table. The signaling information for disaster broadcast may be signaled to this table. This table will be referred to as an

Emergency Alert Table (EAT).

[00532] Hereinafter, a description will be given of signaling information for the disaster broadcast using an FIC chunk. The disaster broadcast may be a specific channel or a text indicating the disaster status. In this embodiment, a message containing text-type disaster information for informing the disaster status is referred to as an emergency alert message for simplicity of description.

[00533] As an example, the signaling information signaled to the FIC chunk contains at least one of emergency alert system (EAS) indication information (or referred to as an EAS indicator field) for informing the current mobile broadcast of existence of an emergency alert message and wake-up indication information (or referred to as a wake-up indicator field) for supporting a wake-up function.

[00534] The EAS indication information informs of existence of an emergency alert message in the current mobile broadcast.

[00535] The wake-up indication information informs of whether the emergency alert message needs the wake-up function. That is, the wake-up indication information is needed to support the wake-up function of the broadcast receiver in issuing a disaster alert.

[00536] Herein, the wake-up function refers to a function for compulsorily switching the broadcast receiver to the active mode broadcast receiver even if the broadcast receiver is in the sleep mode (or referred to as the standby mode when a serious emergency alert message requiring switch to the active mode is issued. To support the wake-up function, the broadcast receiver needs to consistently watch broadcast signals even in the sleep mode and to recognize how urgent the disaster alert is as quickly as possible.

[00537] The FIC chunk is configured with an FIC chunk header and an FIC chunk payload, and the FIC chunk is divided into a plurality of FIC segment payloads. When the FIC segment header

is added to each of the FIC segment payloads, one FIC segment is constructed. The FIC segment is transmitted through one data group. That is, the FIC segment is the smallest unit of transmission at the physical layer.

**[00538]** FIG. 39 illustrates another embodiment of the syntax structure of the FIC segment header to which the EAS indication information and the wake-up indication information are signaled. Details of the FIC segment header of FIG. 39 are the same as those of the FIC segment header of FIG. 36 except that the FIC segment header of FIG. 39 allocates the wake-up indicator field and the EAS indicator field using the two reserved bits of FIG. 36. Accordingly, a description of the fields of FIG. 39 identical to those of FIG. 36 will be omitted.

**[00539]** In FIG. 39, the wake-up indicator field has, for example, 1 bit allocated thereto, and indicates wake-up. The broadcast receiver determines whether to switch to the active mode according to the value of the wake-up indicator field.

**[00540]** In addition, the EAS indicator field has, for example, 1 bit allocated thereto, and indicates existence of an emergency alert message in the current mobile broadcast.

**[00541]** By adding the EAS indicator field to the FIC segment header as described above, the broadcast receiver may quickly sense whether an emergency alert message is included in the current broadcast service.

**[00542]** FIG. 40 illustrates another embodiment of the syntax structure of the FIC segment header to which the EAS indication information and the wake-up indication information are signaled. Details of the FIC segment header of FIG. 40 are the same as those of the FIC segment header of FIG. 36 except that the FIC segment header of FIG. 40 allocates the wake-up indicator field using the two reserved bits of FIG. 36. Accordingly, a description of the fields of FIG. 39 identical to those of FIG. 36 will be omitted.

[00543] That is, in FIG. 40, the wake-up indicator field has, for example, 2 bits allocated thereto, and indicates wake-up. The broadcast receiver determines whether to switch to the active mode according to the value of the wake-up indicator field when an emergency alert message is issued.

[00544] FIG. 41 shows an example of meanings allocated to the values of the 2 bit wake-up indicator field. For example, when the value of the wake-up indicator field is 01, the wake-up function may be enabled. When 10, the wake-up function may be disabled. For example, when the value of the wake-up indicator field is 01, and the current mode is the sleep mode, the broadcast receiver switches to the active mode.

[00545] The opposite case is also possible. That is, when the value of the wake-up indicator field is 01, the wake-up function may be disabled. When 10, the wake-up function may be enabled. In the case that 1 bit is allocated to the wake-up indicator field, the wake-up function may be turned on/off using the 1 bit. Since this is an item selected by the designer, embodiments of the present invention are not limited to the above example.

[00546] While the wake-up indicator of FIG. 39 is provided with 1 bit allocated thereto, the wake-up indicator of FIG. 40 has 2 bits allocated to indicate wake-up.

[00547] In the case that the wake-up indicator field is allocated as shown in FIG. 40, the EAS indication information is allocated, for example, using the reserved bits of the FIC chunk payload.

[00548] FIG. 42 illustrates an embodiment of the syntax structure of the FIC chunk payload to which the EAS indication information is signaled. Details of the FIC chunk payload of FIG. 42 are the same as those of the FIC chunk payload of FIG. 35 except that the EAS indicator field is allocated using the reserved bit. Accordingly, a description of the fields of FIG. 42 identical to those of FIG. 35 will be omitted.

[00549] That is, in FIG. 42, 1 bit is allocated to the ensemble loop for the EAS indicator field (EAS\_ensemble\_indicator). This field indicates existence of an emergency alert message in the current broadcast.

[00550] For example, when the value of the EAS indicator field is 1, an ensemble identified by the ensemble identifier in the ensemble loop may be set to indicate that an emergency alert message has been received. The opposite case is also possible. That is, when the value of the EAS indicator field is 0, the ensemble may be set to indicate that the emergency alert message has been received. Since this is an item selected by the designer, embodiments of the present invention are not limited to the above example.

[00551] In performing wake-up, the broadcast receiver needs to specify an ensemble frequency (or a base band) at which an emergency alert message can be received. To this end, the EAS indicator field is needed.

[00552] That is, informing existence of an emergency alert message using the FIC segment header is the most efficient method for the wake-up function and quick acquisition of an emergency alert message in the broadcast receiver. However, if the FIC segment is the basic unit for creation of an FIC chunk, and the broadcast receiver does not respond to the change of reserved bits of the FIC segment header, existence of an emergency alert message in the current mobile broadcast cannot be recognized once the FIC chunks are collected. Accordingly, in this embodiment, by allocating the EAS indicator field to the FIC chunk payload, the broadcast receiver is allowed to notice whether an emergency alert message exists in the current ensemble immediately after the FIC chunks are collected.

[00553] In addition, in the case that the broadcast receiver is not in the active mode, the broadcast receiver is allowed to receive an emergency alert message with the minimum energy

maintained by signaling wake-up information to the TPC of the smallest unit for signaling.

**[00554]** The wake-up information includes, for example, wake-up indication information. The wake-up information may further include wake-up version information.

**[00555]** For example, 1 bit of the wake-up indication information and 5 bits of the wake-up version information are signaled using the reserved bits in the syntax of the TPC data of FIG 31.

**[00556]** Since the wake-up indication information (wake\_up\_indicator) has been described in detail in the description of the FIC, a detailed description thereof will be omitted. For example, if the value of the wake\_up\_indicator field is 0, and the broadcast receiver is in the sleep mode (or standby mode), the broadcast receiver is compulsorily switched to the active mode. If the value of the field is 1, the broadcast receiver maintains the current mode. That is, in the sleep mode, the receiver is allowed to continue monitoring the TPC. In the case that the receiver is providing the mobile broadcast service, it is allowed to keep providing the mobile broadcast service.

**[00557]** The wake-up version information (wake\_up\_version\_number) indicates the version number of wake-up signaling. The receiver may compare the wake-up version numbers to determine whether the information is new wake-up information before switching from the sleep mode to the active mode to receive the FIC.

**[00558]** By signaling the wake-up information to the TPC data, the broadcast receiver is allowed to determine whether to perform the wake-up function based on the TPC data, namely, whether to compulsorily switch from the current mode to the active mode. In addition, by signaling the wake-up information to the TPC data, the broadcast receiver is allowed to compulsorily perform the wake-up function to receive disaster

alert over mobile broadcast even when failing to receive signaling information related to the disaster alert. Particularly, by signaling the wake-up information to the TPC data, the broadcast receiver is allowed to receive disaster information with the minimum consumption of the battery even in a mode other than the active mode.

**[00559]** Meanwhile, an emergency alert message containing disaster information (or content) may be transmitted in various forms. For example, the emergency alert message may be transmitted in the form of a CAP message or syntax.

**[00560]** Herein, the common alerting protocol (CAP) message is an emergency alert message of a common format created by authoring authorities for generation of a disaster alert. A CAP message generated as above is delivered to various disaster systems including a commercial mobile alert system (CMAS) over the Open Platform for Emergency Networks (OPEN), which is an integrated network for delivery of emergency alert messages. Each disaster alert system processes a CAP message for its own purpose and use, thereby providing a corresponding disaster alert service. For example, the CMAS selects a message proper for a provider network among delivered emergency messages and extracts a text in 90 words or less from the message, thereby providing a disaster service in the form of a text message.

**[00561]** In an embodiment, the CAP message is carried in a descriptor and the descriptor is included in the SMT. The descriptor containing the CAP message will be referred to as an EAS message descriptor (EAS\_Message\_descriptor()). When the EAS descriptor is transmitted, it may be contained in the ensemble level or the service level of the SMT.

**[00562]** FIG. 43 illustrates the syntax structure of the EAS message descriptor according to one embodiment of the present invention.

**[00563]** In FIG. 43, the descriptor\_tag field (8bits) is a

descriptor identifier. An identifier (or a tag value) to identify whether a descriptor is the EAS message descriptor is allocated.

**[00564]** The descriptor\_length field (8 bits) indicates the remaining length of this descriptor from the descriptor\_length field to the end of the descriptor in unit of byte.

**[00565]** The Length\_of\_CAP field (8 bits) indicates the length of a CAP message transmitted in the multiple string structure.

**[00566]** The CAP message in the multiple string structure is transmitted by repeating the CAP field by the value of the Length\_of\_CAP field. At this time, a separate EAS syntax may be transmitted in place of the CAP message.

**[00567]** FIG. 44 illustrates the syntax structure of the EAS message descriptor according to another embodiment of the present invention.

**[00568]** In FIG. 44, the descriptor\_tag field (8bits) is a descriptor identifier, and an identifier (or a tag value) to identify whether a descriptor is the EAS message descriptor is allocated to this field.

**[00569]** The descriptor\_length field (8 bits) indicates the remaining length of this descriptor from the descriptor\_length field to the end of the descriptor in the unit of byte.

**[00570]** The CAP\_version\_number field (4bits) indicates the version information of a CAP message. For example, when the CAP message changes, the value of the CAP\_version\_number field increases. Accordingly, the broadcast receiver is allowed to determine whether the CAP messages are repeated using this field value. That is, since the CAP message is emergency information, the same content may be periodically transmitted. In this case, every time a CAP message is received, the broadcast receiver can determine whether the received message is a change of the previously received message or new information, using the value of the CAP\_version\_number field.

- [00571]** The EAS\_NRT\_included\_flag field (1bit) indicates whether a non-real time (NRT) service related to a current disaster broadcast service (or emergency alert message) exists.
- [00572]** If the EAS\_NRT\_included\_flag field indicates existence of an NRT service related to the current disaster broadcast, an EAS NRT loop including the essential\_content\_linkage field and num\_associated\_service field exists.
- [00573]** The essential\_content\_linkage field (8 bits) indicates the value of linkage between a file (or a content item) sent in NRT and the content (or a content item) in the ESG. For example, if the value of the essential\_content\_linkage field is 0, the user interface (UI) displays a list of NRT services for the EAS. If the value is not 0, the UE displays a file having the value of the essential\_content\_linkage field as the value of content\_linkage on the screen. This means that the content contained in the file provided in NRT is highly associated with a disaster broadcast (or emergency alert message).
- [00574]** The num\_associated\_service field (4 bits) indicates the number of services associated with the current disaster broadcast and transmitted in NRT. An identifier of each of the associated services is indicated by repeating the associated\_service\_id field (16 bit) by the value of the num\_associated\_service field.
- [00575]** The CAP\_message\_type field (1bit) indicates whether an emergency alert message is transmitted in the CAP message structure on syntax structure.
- [00576]** The length\_of\_CAP\_message\_id field (7 bits) indicates the length of the text of the CAP message identifier (ID).
- [00577]** The CAP\_message\_id field (variable) indicates the CAP message identifier in a text.
- [00578]** The length\_of\_CAP field (16bit) indicates the length

of the CAP message.

**[00579]** The CAP field (variable) contains a CAP message or EAS syntax.

**[00580]** Since the same content of the EAS message descriptor shown in FIG. 43 or 44 should be broadcasted in all broadcasts, and therefore the EAS message descriptor may be included in at least one of the ensemble level and service level of the SMT. The SMT is essentially included in all ensembles of a channel. That is, the EAS message descriptor may be transmitted as the ensemble level descriptor of the SMT, or as the service level descriptor. FIG. 45 illustrates an example of transmission of the EAS message descriptor included in the ensemble level, and FIG. 46 illustrates an example of transmission of the EAS message descriptor included in the service level.

**[00581]** In the case that the EAS message descriptor is transmitted through the ensemble level descriptor as shown in FIG. 45, the content contained in the EAS message descriptor is applied to all services (e.g., TV service #1 and TV service #2) of a corresponding ensemble (e.g., ensemble #1) in the same manner. If the EAS message descriptor is contained in the ensemble level and transmitted, the necessity of sending the EAS message descriptor for each of the services of the SMT is eliminated. Accordingly, the size of the SMT may be reduced. When the EAS message descriptor is transmitted as the ensemble level descriptor, the value of EAS\_NRT\_included\_flag field shown in FIG. 44 may be set to 0. In this case, NRT associated information (i.e., disaster-related additional information) cannot be transmitted except the CAP message.

**[00582]** In the case that the EAS message descriptor is transmitted as the service level descriptor as shown in FIG. 46, the same content of the EAS message descriptor is contained in each service of the SMT. This may increase the size of the SMT compared to the case of transmission of the descriptor as the ensemble level descriptor, but can

advantageously allow transmission of the NRT service-associated information to express additional information associated with the emergency alert message (i.e., disaster-related additional information) since all the descriptors of the SMT used in NRT are in the service level.

**[00583]** When the EAS message descriptor is transmitted as the service level descriptor, the value of the EAS\_NRT\_included\_flag field shown in FIG. 44 may be set to one of 0 and 1. If the value of the EAS\_NRT\_included\_flag field is 0, it indicates that there is no disaster-related additional information which is transmitted in NRT. If the value is 1, it indicates that there is disaster-related additional information which is transmitted in NRT.

**[00584]** The broadcast receiver extracts the EAS message descriptor by parsing the received SMT included in the ensemble, processes the CAP message contained in the EAS message descriptor, and then displays disaster information in the form of a text at the lower end 601 of the screen as shown in FIG. 47. Herein, the EAS message descriptor may be included in the ensemble level of the SMT or in the service level. In addition, while the disaster information in the form of a text is illustrated as being displayed at the lower end of the screen, it may be displayed at another portion of the screen, for example, one of the upper end, left side and right side. In addition, the disaster information in the form of a text may be scrolled from one side to the other.

**[00585]** As described above, the disaster-related additional information broadcast according to embodiments of the present invention may be transmitted in NRT. Herein, the disaster-related additional information includes various types of information such as image, video, and text, which may be greatly useful for the user when disaster happens.

**[00586]** FIG. 48 illustrates displaying the received disaster information in the form of a text contained in the EAS message

descriptor at the lower end 601 of the screen and displaying the disaster-related additional information received in NRT in the form of an image or text at other portions 602 to 604 of the screen at the same time. The screen shown in FIG. 48 is simply an example. The screen size and the disposition of the disaster-related additional information may vary depending on the types and the number of pieces of the disaster-related additional information.

**[00587]** In one embodiment, the disaster-related additional information may be configured in one file and transmitted in NRT.

**[00588]** In mobile broadcast, the real time (RT) service literally refers to a service provided in real time. That is, this service is a time-dependent service. On the other hand, the NRT service refers to a service provided in non-real time compared to the RT service. That is, the non-real time service is not time-dependent. For example, a broadcasting station may transmit an RT service and provide an NRT service using a remaining bandwidth or a dedicated bandwidth. That is, the RT service and the NRT service are transmitted over the same channel or different channels.

**[00589]** In another example, the broadcast receiver according is capable of receiving a non-real time (NRT) service through media such as terrestrial broadcast, cable, and the Internet.

**[00590]** The NRT service is stored in the storage medium of the broadcast receiver, and is then displayed on a display device at a predetermined time or according to a request from the user. The NRT service may be stored in the form of a file in the storage. In one embodiment, the storage medium may be a built-in HDD mounted in the broadcast receiver. In another embodiment, the storage medium may be an external HDD or a universal serial bus (USB) memory connected to the exterior of a broadcast receiving system.

**[00591]** In one embodiment, one NRT service includes at least

one content item (or referred to as content or NRT content), and one content item includes at least one file. In this present invention, the 'file' has the same meaning as the term 'object'. A content item is the smallest unit that can be independently reproduced. For example, assume that news is provided in non-real time and that the new includes news on the economy, politics and life. In this case, the news may be viewed as the NRT service, and each of the news on the economy, politics, and life may be viewed as a content item. each of the news on the economy, politics, and life is configured in at least one files.

**[00592]** Content items/packets configuring an NRT service at the transmitter side such as a broadcasting station are packetized according to the file transfer protocol, and then packetized according to the asynchronous layered coding/layered coding transport (ALC/LCT) scheme. The packetized ALC/LCT data is then packetized according to the UDP, and the packetized ALC/LCT/UDP data is in turn packetized according to the IP scheme to form ALC/LCT/UDP/IP data. The packetized ALC/LCT/UDP/IP data will be referred to as IP datagram for simplicity of the description of the present invention. The OMA BCAST service guide information may also undergo the same process as for the content item/files to configure an IP datagram.

**[00593]** The IP datagrams for the NRT service are contained in at least one RS frame belonging to an ensemble (or referred to as a subnet) when they are transmitted. An IP datagram of the OMA BCAST service guide may also be transmitted in the RS frame. In addition, similar to the mobile service, access information for accessing the NRT service is signaled to the SMT. That is, the SMT provides RT service or NRT service and access information of the components (or content items) contained in each service.

**[00594]** Accordingly, if the service contained in the mobile

broadcast is the NRT service, signaling information including the access information of the FLUTE session, which carries content items/files configuring the NRT service, may be extracted from the SMT. In addition, detailed information on content items configuring the NRT service may be extracted from the OMA BCAST service guide (SG) information.

**[00595]** At this time, the SMT contains one NRT service descriptor and one or more capabilities descriptor for each NRT service in the service level.

**[00596]** The NRT service descriptor indicates whether an NRT service exists, identifies the consumption model of the NRT service, and provides other optional information about the NRT service. That is, the NRT service descriptor describes information items indicating that the service described in the service level of the SMT is an NRT service.

**[00597]** The capabilities descriptor provides a list of capabilities used for the NRT service or a content item (e.g., download protocol, FEC algorithm, wrapper/archive formats, compression algorithm, and media type). In addition, a nonessential protocol may be listed. That is, the capabilities descriptor describes the capabilities of the files configuring the NRT service.

**[00598]** Meanwhile, for the broadcast receiver to process disaster-related additional information received in NRT and provide the processed information to the user, information allowing identifying the files received in NRT as disaster-related additional information is needed.

**[00599]** In one embodiment of the present invention, the information for identifying a file as disaster-related additional information is signaled to at least one of the NRT service descriptor and the capabilities descriptor transmitted to the service level of the SMT.

**[00600]** FIG. 49 illustrates the syntax structure of a capabilities descriptor according to one embodiment of the

present invention.

- [00601]** In FIG. 49, the descriptor\_tag field (8bits) is a descriptor identifier. An identifier (or a tag value) to identify whether a descriptor is the capabilities descriptor is allocated to this field.
- [00602]** The descriptor\_length field (8 bits) indicates the remaining length of this descriptor from the descriptor\_length field to the end of the descriptor in unit of byte.
- [00603]** The capability\_code\_count field (8 bit) indicates the number of values of capability\_code. The essential\_indicator field and capability\_code field are repeated by a value of the capability\_code\_count field.
- [00604]** The essential\_indicator field (1 bit) indicates whether supporting the capability indicated by capability\_code is essential for meaningful existence of the NRT service or content item.
- [00605]** The capability\_code field (7 bits) indicates a specific capability. If the value of the capability\_code field is greater than 0x6F, a format\_identifier field indicating an format identifier exists.
- [00606]** The capability\_string\_count field (8 bits) indicates the number of values of the capability\_string. The essential\_indicator field, capability\_category\_code field, capability\_string\_length field, and capability\_string() field are repeated by the value of the capability\_string\_count field .
- [00607]** The essential\_indicator field (1 bit) indicates whether supporting the capability indicated by capability\_string is essential for meaningful existence of the NRT service or content item.
- [00608]** The capability\_category\_code field (7 bits) indicates the capability category of a subsequent string value.
- [00609]** The capability\_string\_length field (8 bits) indicates the length of a subsequent capability\_string() field.

- [00610] The capability\_string() field is a string (i.e., a character string) including the capability.
- [00611] In one embodiment, the capability\_string() field, which is capable of classifying files, is used to indicate that a file is related to disaster-related additional information.
- [00612] That is, by allocating a new string value (e.g., x-application/eas) to the value of the capability\_string() field, the received files with the "x-application/eas" capability is identified as containing disaster-related additional information.
- [00613] FIG. 50 illustrates the syntax structure of an NRT service descriptor according to one embodiment of the present invention.
- [00614] In FIG. 50, the descriptor\_tag field (8 bits) is a descriptor identifier. An identifier (or a tag value) to identify whether a descriptor is the NRT service descriptor is allocated to this field.
- [00615] The descriptor\_length field (8 bits) indicates the remaining length of this descriptor from the descriptor\_length field to the end of the descriptor in unit of byte.
- [00616] The consumption\_model field (6 bits) signals an intended consumption model for an NRT service associated with the descriptor.
- [00617] The auto-update field (1 bit) indicates whether to provide an option for automatic updating of the NRT service to the user.
- [00618] The storage\_reservation\_present field (1 bit) indicates existence of the storage\_reservation field.
- [00619] The default\_content\_size\_present field (1 bit) indicates existence of the default\_content\_size field.
- [00620] The storage\_reservation field (24bit) indicates the minimum storage capacity of the receiver required to store a content item transmitted through the NRT service.

**[00621]** The default\_content\_size field (40bit) indicates a default size for the content item.

**[00622]** In one embodiment of the present invention, the consumption model field indicating a description of a scheme in which an NRT service is provided is used to indicate that the NRT service is related to the disaster-related additional information.

**[00623]** That is, a currently reserved value is allocated to the consumption model field, and the files including a NRT service having this value are identified as containing the disaster-related additional information.

**[00624]** FIG. 51 illustrates meanings of the values allocated to the consumption model field. FIG. 51 illustrates an example of allocating 0x04 to the consumption model field in the case that the NRT service is related to the disaster-related additional information.

**[00625]** As described above, by signaling, to NRT service descriptor and/or capabilities descriptor among the descriptors transmitted in the service level of the SMT, the information for identifying whether the files included in the NRT service contains disaster-related additional information broadcast, the disaster-related additional information may be efficiently provided using NRT.

**[00626]** According to one embodiment, the transmitter side may describe the information delivering the ESG corresponding to the NRT service in the GAT, and the broadcast receiver may combine this information and the SMT and download the ESG. The GAT includes information about a provider sending the ESG. At this time, separately sending the ESG information associated with the EAS is more effective for quick transmission of the disaster information. In this case, to distinguish the ESG from an ESG for the existing SG data, the provider name may be allowed to represent the EAS.

**[00627]** FIG. 52 is a flowchart illustrating a method of the

broadcast receiver receiving and displaying an emergency alert message containing disaster information and disaster-related additional information according to one embodiment of the present invention.

**[00628]** In FIG. 52, the emergency alert message containing disaster information may be carried in the EAS message descriptor of FIG. 43 or 44 and received, and the disaster-related additional information may be received in the form of a file in NRT. The information identifying whether the file is the disaster-related additional information may be received by being signaled to the capabilities descriptor of FIG. 49 or the NRT service descriptor of FIG. 50. Herein, the EAS message descriptor may be one of the ensemble level descriptor and service level descriptor of the SMT, and each of the capabilities descriptor and NRT service descriptor is one of the service level descriptors of the SMT.

**[00629]** That is, the mobile broadcast is tuned by a tuner of the broadcast receiver (S651), and then a service signaling channel included in the tuned mobile broadcast is acquired (S652). Herein, the service signaling channel may be encapsulated in the IP datagrams having a well-known IP destination address and a well-known destination UDP port number. In other words, all the IP datagrams (i.e., UDP/IP packets) transmitted through the service signaling channel have the same well-known target IP address and the same well-known target UDP port number. For example, when the SMT, RRT, and GAT are assumed to be transmitted through the service signaling channel, all UDP/IP packets transmitting the SMT, RRT, and GAT have the same target IP address and the same target UDP port number. In addition, the target IP address and target UDP port number are well-known values, namely, values known to the receiving system according to agreement by the transmitting/receiving system. Accordingly, signaling tables contained in the service signaling channel may be

distinguished from each other by a table identifier. The table identifier may be `table_id` present in a signaling table or the header of a signaling table session. If necessary, `table_id_extension` may be further referenced to distinguish the tables from each other.

**[00630]** Information of the signaling tables distinguished by the `table_id` field and/or `table_id_extension` field is stored in the storage unit of the broadcast receiver (S653). In addition, whether the EAS message descriptor exists in the SMT is confirmed (S654). If the EAS message descriptor exists, the emergency alert message received through the EAS message descriptor is processed and disaster information is displayed in the form of a text on a portion of the screen, as shown in FIG. 47 (S655).

**[00631]** In addition, whether the capabilities descriptor or NRT service descriptor exists in the SMT and whether an indication indicating that a file included in the NRT service is the EAS, i.e., disaster-related additional information is signaled to at least one of the capabilities descriptor and the NRT service descriptor are confirmed (S656).

**[00632]** When it is confirmed in step S656 that an indication indicating the disaster-related additional information is signaled, the NRT service is downloaded (S657). Then, whether the expiration time of the files contained in the downloaded NRT service has passed (S658). If it is confirmed that the expiration time has not passed, disaster-related additional information included in at least one file received in NRT is displayed on a portion of the screen, as shown in FIG. 48 (S659). Herein, the disaster-related additional information may be various types of information such as image, video, and text which are related to the disaster information contained in the emergency alert message.

**[00633]** FIG. 53 is a flowchart illustrating a method of the broadcast receiver receiving and displaying an emergency alert

message containing disaster information and disaster-related additional information according to another embodiment of the present invention.

**[00634]** In FIG. 53, the emergency alert message containing disaster information may be carried in the EAS message descriptor of FIG. 44, which include the essential\_content\_linkage field, and received, and the disaster-related additional information may be received in the form of a file in NRT. The information identifying whether the file is the disaster-related additional information may be received by being signaled to the capabilities descriptor of FIG. 49 or the NRT service descriptor of FIG. 50. Herein, the EAS message descriptor may be one of the ensemble level descriptor and service level descriptor of the SMT, and each of the capabilities descriptor and NRT service descriptor is one of the service level descriptors of the SMT.

**[00635]** That is, the mobile broadcast is tuned by a tuner of the broadcast receiver (S661), and then a service signaling channel included in the tuned mobile broadcast is acquired (S662). The details of the service signaling channel are identical to those illustrated with reference to FIG. 52, and thus a description thereof will be omitted.

**[00636]** Information of the signaling tables identified by the table\_id and/or table\_id\_extension field included in each signaling table of the service signaling channel is stored in the storage unit of the broadcast receiver (S663). In addition, whether the EAS message descriptor exists in the SMT is confirmed (S664). If the EAS message descriptor exists, the emergency alert message received through the EAS message descriptor is processed and disaster information is displayed in the form of a text on a portion of the screen, as shown in FIG. 47 (S665).

**[00637]** In addition, whether an NRT service related to the EAS exists is confirmed (S666). Herein, whether an NRT service

related to the EAS exists may be conformed using the num\_associated\_service field of the EAS message descriptor. If one or more NRT service related to the EAS exists, the EAS message descriptor provides an identifier of the NRT service related to the EAS (associated\_service\_id).

**[00638]** In step S666, when existence of one or more NRT service related to the EAS is confirmed, an NRT service is downloaded using the corresponding NRT service identifier (S667). Then, whether the value of the essential\_content\_linkage field of the EAS message descriptor is 0 is confirmed (S668). The essential\_content\_linkage field indicates the value of linkage between a file included in the NRT service and a content item in the ESG. If the value of the essential\_content\_linkage field is 0, the user interface (UI) displays a list of NRT services for the EAS (S669). If the value is not 0, the UE displays a file having the value of the essential\_content\_linkage field as the value of content\_linkage on the screen (S670).

**[00639]** As described above, an emergency alert message containing actual disaster information (or content) may be transmitted in the form of a CAP message or syntax.

**[00640]** At this time, an emergency alert message in the form of the CAP or an emergency alert message in the form of syntax may be encapsulated in an IP datagram and transmitted.

**[00641]** That is, as shown in FIG. 54, the IP datagram may be in the form of an IP packet configured with a header and a payload, and the header may be configured with an IP header and a UDP header. In addition, the payload includes an emergency alert message in the form of the CAP or an emergency alert message in the form of syntax. The emergency alert message in the form of syntax included in the payload has the form of a table. This table will be referred to as an EAS table. When the emergency alert message in the form of the CAP is transmitted by the payload, the length of the payload may

be calculated based on the length of the IP/UDP header.

**[00642]** FIG. 55 is a view illustrating the syntax structure of an emergency alert system descriptor (emergency\_alerting\_system\_descriptor ()) transmittable by the payload of FIG. 54. The descriptor may be a session, and the EAS table may be configured using at least one descriptor.

**[00643]** In FIG. 55, the descriptor\_tag field (8bits) is a descriptor identifier. An identifier (or a tag value) to identify whether a descriptor is the emergency alert system descriptor is allocated.

**[00644]** The descriptor\_length field (8 bits) indicates the remaining length of this descriptor from the descriptor\_length field to the end of the descriptor in unit of byte.

**[00645]** The descriptor\_number field (4 bits) indicates the number of the current descriptor among the descriptors included in the EAS table.

**[00646]** The last\_descriptor\_number field (4 bits) indicates the number of the last descriptor among the descriptors included in the EAS table.

**[00647]** The EAS\_event\_id field (16 bits) indicates an identifier of an EAS event.

**[00648]** The EAS\_originator\_code field (24 bits) indicates an entity having activated the EAS.

**[00649]** The EAS\_event\_code\_length field (8 bits) indicates the length of the EAS\_event\_code field.

**[00650]** The EAS\_event\_code field (variable) indicates an EAS event, i.e., a type of the EAS. For example, a type of the emergency alert such as flood, earthquake and terrorism may be indicated along with the importance of a specific emergency such as large earthquake, medium earthquake, and small earthquake.

**[00651]** The alert\_message\_time\_remaining field (8 bits) indicates the remaining output time, of an emergency alert message.

- [00652] The event\_start\_time field (32 bits) indicates the start time of display of the EAS event.
- [00653] The event\_duration field (16 bits) indicates the duration time of display of the EAS event.
- [00654] The alert\_priority field (4 bits) indicates the priority or importance of the EAS event. That is, depending on the value of the alert\_priority field, processing of the received emergency alert message is determined. In other words, whether to unconditionally ignore the emergency alert message or ignore the same under certain conditions, or to compulsorily tune the message to a specific broadcast channel is determined.
- [00655] The EAS\_major\_channel\_number field (8 bits) indicates a major channel number related to the broadcast channel.
- [00656] The EAS\_minor\_channel\_number field (8 bits) indicates a minor channel number related to the broadcast channel.
- [00657] The alert\_text\_length field (16 bits) indicates the total number of bytes of the alert\_text() field.
- [00658] The alert\_text() field (variable) includes disaster information to be displayed in the form of a text.
- [00659] The location\_code\_count field (8 bits) counts region definitions following the 'for loop' syntax.
- [00660] The state\_code field (8 bits) indicates a state or territory associated with an emergency alert. The field may have value between 0 and 99.
- [00661] The county\_subdivision field (4 bits) may have a value between 0 and 9 which defines the county division (state subdivision).
- [00662] The county\_code field (10 bits) expresses a specific county associated with the emergency alert among other states. The field may have a value between 0 and 999.
- [00663] According to one embodiment of the present invention,

the emergency alert message may be transmitted at the service level or component level. In the case that the emergency alert message is transmitted at the service level, the emergency alert message is considered as one service, which will be referred to as the EAS service for ease of description. In addition, for the broadcast receiver to support the EAS service, signaling of information to identify the EAS service (hereinafter, referred to as EAS service identifying information) is needed.

**[00664]** In one embodiment of the present invention, one of the reserved values of the MH\_service\_category field in the service loop of the SMT is used for the EAS service identifying information. According to one embodiment, if a service in the service loop of the SMT is an EAS service, 0x04 may be allocated as the value of the MH\_service\_category field, as shown in FIG. 56. In this case, the broadcast receiver may recognize that signaling information of the service described in the service loop of the SMT is the signaling information of the EAS service. Herein, 0x04 allocated as the value of the MH\_service\_category field is simply an example, any category value between 0x00 and 0xFF which is not used by existing services may be allocated. To identify the EAS service, the MH\_service\_id field may be further used.

**[00665]** In the case that the emergency alert message is transmitted by a separate service, transmitting signaling once for each ensemble is sufficient, and therefore this case is advantageous in view of use of a bandwidth.

**[00666]** Meanwhile, in the case that the emergency alert message is transmitted at the component level, the emergency alert message is considered as one component just as a audio component and a video component. This component will be referred to as an EAS component for ease of description. For the broadcast receiver to support the EAS component, signaling of information to identify an EAS component (hereinafter,

referred to as EAS component identifying information) is needed.

**[00667]** In one embodiment of the present invention, one of the reserved values of the component\_type field in component\_descriptor(), shown in FIG. 38, among the component level descriptors of the SMT is used for the EAS component identifying information. For example, if a component in the component loop of the SMT is the EAS component, 43 is allocated as the value of the component\_type field. In this case, the broadcast receiver may recognize that the signaling information of the component described in the component loop of the SMT is the signaling information of the EAS component. Herein, allocation of 43 to the component\_type field is simply an example. Any value between 43 and 71 that is not used by the existing components may be allocated.

**[00668]** The component\_type field in component\_descriptor() indicates a value for identifying the encoding format of a component. This identification value may be one of the values allocated for payload\_type of the RTP/AVP stream, one of the values predetermined according to agreement by the transmitter/receiver side, or a dynamic value between 96 and 127. For the components configuring media transmitted via the RTP, a value of the component\_type field needs to be identical to the value of payload\_type in the RTP header of an IP stream which transmits a corresponding component.

**[00669]** In addition, the component\_data (component\_type) field, which varies depending upon the value of the component\_type field, provides encoding parameters and/or other parameters which are necessary for rendering of a corresponding component. That is, the structure of an component\_data element is determined by the value of the component\_type field.

**[00670]** For example, if the value of the component\_type field is 35, the component\_data() field provides component

data for an H.264/AVC video stream.

**[00671]** If the component\_type field is 38, the component\_data() field provides data for FLUTE file delivery.

**[00672]** If the component\_type field is 43, the component\_data() field provides signaling information for an emergency alert message, as shown in FIG. 57.

**[00673]** FIG. 57 illustrates an example of the bit stream syntax structure of component\_data(), which provides signaling information for an emergency alert message. Fields of the component\_data() are described below.

**[00674]** In FIG. 57, the EAS\_event\_ID field (16 bits) indicates a unique identifier of an EAS event corresponding to the emergency alert message.

**[00675]** The payload\_type field (2 bits) indicates whether an emergency alert message transmitted in the form of an IP datagram is of the CAP type or the syntax type. In this case, the broadcast receiver is allowed to identify, using the payload\_type field, whether the received emergency alert message is CAP type data or syntax type data. That is, the broadcast receiver is allowed to identify the data type of the emergency alert message at the component level.

**[00676]** In another embodiment of the present invention, by allocating the payload\_type field to the reserved bits in the component level of the SMT, the data type of the emergency alert message may be identified at the component level.

**[00677]** In another embodiment of the present invention, by allocating the payload\_type field to the reserved bits in the service level of the SMT, the data type of the emergency alert message may be identified at the service level.

**[00678]** The status field (6 bits) indicates the status of an emergency alert message.

**[00679]** The urgency field (8 bits) indicates the degree of urgency of a disaster contained in the emergency alert message.

**[00680]** The severity field (8 bits) indicates the degree of

severity of a disaster contained in the emergency alert message.

[00681] The certainty field (8 bits) indicates the degree of certainty of a disaster contained in the emergency alert message.

[00682] The event\_start\_time field (32 bits) indicates the start time of display of a disaster alert contained in the emergency alert message.

[00683] The event\_duration field (16 bits) indicates the duration time of a disaster alert contained in the emergency alert message.

[00684] Meanwhile, in one embodiment, a new table is defined to signal signaling information for an emergency alert message. This table will be referred to as an emergency alert table (EAT) in this embodiment.

[00685] An IP datagram including the EAT may be transmitted through a service signaling channel.

[00686] In the case that the EAT is transmitted through the service signaling channel, the EAS\_ensemble\_indicator field included in the FIC chunk payload of FIG. 42 may indicate whether the EAT is transmitted through the service signaling channel of a corresponding ensemble. For example, if the value of the EAS\_ensemble\_indicator field is 1, it may indicate that the EAT is transmitted through the service signaling channel included in the ensemble. If the value is 0, it may indicate that the EAT is not transmitted.

[00687] The EAT may provide access information of an emergency alert message transmitted in the form of an IP datagram. The access information may include an IP address and a UDP port number.

[00688] The service signaling channel through which the EAT is transmitted may further include a signaling table (i.e., at least one of the SMT, GAT, RRT, SLT, and CIT) as well as the EAT. At this time, IP datagrams of the service signaling

channel may have the same well-known IP address and well-known UDP port number. Accordingly, the EAT included in the service signaling channel is identified by the table identifier. That is, the table identifier may be `table_id` existing in the header of the corresponding table or corresponding table section. If necessary, `table_id_extension` may be further referenced..

**[00689]** FIG. 58 illustrates the bit stream syntax structure of an EAT section according to one embodiment of the present invention. While the EAT section is illustrated as being drafted in the form of an MPEG-2 private section, the format of the data in the EAT section may have any form. At this time, whether the EAT is configured with one section or a plurality of sections may be confirmed through the `table_id` field, `section_number` field, and `last_section_number` field in the EAT section. In addition, a corresponding table may be completed by removing IP headers and UDP headers of the IP datagrams from the service signaling channel and collecting sections having the same table identifier. For example, when the sections having the table identifier allocated to the EAT, one EAT may be completed.

**[00690]** Hereinafter, the fields transmittable through the EAT section will be described.

**[00691]** The `table_id` field (8 bits) is a field to identify the type of the table. Through this field, it may be confirmed that the table is the EAT.

**[00692]** The `section_syntax_indicator` field (1bit) is an indicator to define the section form of the EAT. The section format may be, for example, short-form syntax ('0') of MPEG.

**[00693]** The `private_indicator` field (1bit) indicates whether the EAT follows the private section.

**[00694]** The `section_length` field (12bits) indicates the section length of the remaining EAT subsequent to a corresponding field.

- [00695] The `table_id_extension` field (16bits) is table-dependent, and is a logical portion of the `table_id` field which provides the range of the remaining fields. The `table_id_extension` field includes a `EAT_protocol_version` field and a `ensemble_id` field.
- [00696] The `EAT_protocol_version` field (8 bits) indicates a protocol version to allow the EAT that parameters having a different structure over those defined in a current protocol transmit.
- [00697] The `ensemble_id` field (8 bits) is an ID value related to the corresponding ensemble, and a value between 0x00 and 0x3F may be allocated thereto. Preferably, the value of this field is drawn from the `parade_id` of TPC data. In the case that the ensemble is transmitted through a primary RS frame, the most significant bit (MSB) is set to '0', and the remaining 7 bits are used as a value of `parade_id` of a corresponding parade. In the case that the ensemble is transmitted through a secondary RS frame, the MSB is set to '1', and the remaining 7 bits are used as a value of `parade_id` of a corresponding parade.
- [00698] The `version_number` field (5bits) indicates the version number of the EAT.
- [00699] The `current_next_indicator` field (1 bit) indicates whether the EAT section is currently applicable.
- [00700] The `section_number` field (8 bits) indicates the number of a current EAT section.
- [00701] The `last_section_number` field (8 bits) indicates the number of the last section configuring the EAT.
- [00702] The `num_MH_services` field (8 bits) indicates the number of mobile services in an EAT section.
- [00703] By performing 'for' loop (or referred to as mobile service loop) by the number of mobile services corresponding to the value of the `num_MH_services` field, signaling information about a plurality of mobile services is provided.

That is, signaling information about each of the mobile services is indicated for each of the mobile services included in the EAT section. At this time, field information may be provided for each mobile service as follows.

**[00704]** The MH\_service\_id field (16 bits) indicates a value that uniquely identifies a mobile service.

**[00705]** The MH\_service\_status field (2 bits) identifies the status of the mobile service. Herein, the MSB indicates whether the mobile service is active ('1') or non-active ('0'), and the LSB indicates whether the mobile service is hidden ('1') or not ('0').

**[00706]** The SP\_indicator field (1 bit) indicates whether service protection of the mobile service is applied. If the value of the SP\_indicator field is 1, the service protection is applied at least one of the components required for the mobile service to provide meaningful presentation mobile service.

**[00707]** The urgency field (8 bits) indicates the degree of urgency of an emergency alert message contained in the mobile service.

**[00708]** The severity field (8 bits) indicates the degree of severity of the emergency alert message contained in the mobile service.

**[00709]** The certainty field (8 bits) indicates the degree of certainty of the emergency alert message contained in the mobile service.

**[00710]** The event\_start\_time field (32 bits) indicates the start time of display of the emergency alert message contained in the mobile service.

**[00711]** The event\_duration field (16 bits) indicates the duration time of display of the emergency alert message contained in the mobile service.

**[00712]** In the case that the IP\_version\_flag field (1 bit) is set to '1', it indicates that the service\_source\_IP\_address

field and service\_destination\_IP\_address field are IPv6 addresses. In the case that the IP\_version\_flag field (1 bit) is set to '0', it indicates that the service\_source\_IP\_address field and service\_destination\_IP\_address field are IPv4 addresses.

**[00713]** In the case that the service\_source\_IP\_address\_flag field (1bit) is set to '1', it indicates that a value of the source IP address for the service exists.

**[00714]** In the case that the service\_destination\_IP\_address\_flag field (1bit) is set to '1', it indicates that a value of the destination IP address for the service exists.

**[00715]** The IP\_payload\_type field (2 bit) indicates whether the emergency alert message transmitted in the form of an IP datagram is of the CAP type or the syntax type.

**[00716]** The service\_source\_IP\_address field (32 or 128 bits) needs to be interpreted when the service\_source\_IP\_address\_flag field is set to '1', but does not need to be interpreted when the service\_source\_IP\_address\_flag field is set to '0'. In the case that the service\_source\_IP\_address\_flag field is set to '1', and the IP\_version\_flag field is set to '0', the above field indicates a 32bit IPv4 address representing the source of the mobile service. In the case that the IP\_version\_flag field is set to '1', the above field indicates a 32bit IPv6 address the source of the mobile service.

**[00717]** The service\_destination\_IP\_address field (32 or 128 bits) needs to be interpreted when the service\_destination\_IP\_address\_flag field is set to '1', but does not need to be interpreted when the service\_destination\_IP\_address\_flag field is set to '0'. In the case that the service\_destination\_IP\_address\_flag field is set to '1', and the IP\_version\_flag field is set to '0', the above field indicates a 32bit destination IPv4 address for the

mobile service. In the case that the service\_destination\_IP\_address\_flag field is set to '1', and the IP\_version\_flag field is set to '1', the above field indicates a 64bit destination IPv6 address for the mobile service.

**[00718]** Meanwhile, if the value of the severity field is 1, i.e., if a disaster is very severe, compulsory tuning information is signaled.

**[00719]** That is, the force\_channel\_num field (8 bits) indicates the channel number to be compulsorily tuned, and the force\_service\_id field (16 bits) indicates a service identifier to be compulsorily tuned.

**[00720]** The num\_MH\_EAS\_descriptor field (8 bits) indicates the number of descriptors that provide additional information of the service level.

**[00721]** A number of MH\_EAS\_descriptor() fields corresponding to the value of the num\_MH\_EAS\_descriptor field are included in the service loop to provide additional information about the emergency alert message.

**[00722]** If the emergency alert message is not transmitted in the form of an IP datagram, the emergency alert message may be included in the MH\_EAS\_descriptor() and transmitted. To this end, the EAT may further include information for identifying whether the emergency alert message is transmitted in the form of an IP datagram or included in the EAT and transmitted.

**[00723]** FIG. 59 is a flowchart illustrating a method of the broadcast receiver receiving and displaying an emergency alert message containing disaster information according to one embodiment of the present invention.

**[00724]** In FIG. 59, the emergency alert message containing disaster information may be received in the form of an IP datagram, and the signaling information including the access information of the emergency alert message may be signaled to the EAT and received.

**[00725]** That is, the mobile broadcast is tuned by a tuner of the broadcast receiver (S701), and then an IP datagram of a service signaling channel and an IP datagram including the emergency alert message are identified through IP filtering (S702). In one embodiment, the IP address and the UDP port number for accessing the IP datagram of the service signaling channel may be a well-known IP destination address and a well-known destination UDP port number, respectively. In addition, the IP address and the UDP port number for accessing the IP datagram of the emergency alert message may be signaled to the EAT. In another embodiment, the IP address and the UDP port number for accessing the IP datagram of the emergency alert message may be signaled to the SMT.

**[00726]** Information of each signaling table of the service signaling channel identified by the `table_id` field and/or `table_id_extension` fields of the signaling table is stored in the storage unit of the broadcast receiver (S703). In addition, whether the EAT exists in the service signaling channel is confirmed (S704). The confirmation may be performed using the `table_id` field and/or `table_id_extension` field of the EAT. If the EAT exists, the emergency alert message in the form of the IP datagram is processed using the information included in the EAT. That is, using the value of the `payload_type` field of the EAT, whether the data received with a payload of the IP datagram is a CAP type emergency alert message of a syntax type emergency alert message is confirmed (S705). If the data is a CAP type emergency alert message, a CAP parser (not shown) processes the emergency alert message (S706). If the data is a syntax type emergency alert message, a syntax parser (not shown) processes the emergency alert message (S707). Then, the processed message is displayed in the form of a text on a portion of the screen as shown in FIG. 47 (S708).

**[00727]** In addition, whether the capabilities descriptor or NRT service descriptor exists in the SMT and whether an

indication indicating that a file included in the NRT service is the EAS, i.e., disaster-related additional information is signaled to at least one of the capabilities descriptor and the NRT service descriptor are confirmed (S656).

**[00728]** Herein, at least one file including the additional information related to the emergency alert message (i.e., disaster-related additional information) may be received in NRT, and the signaling information for the disaster-related additional information may be signaled in the service level or component level of the SMT. Then, the received file is processed in NRT based on the signaling information of the disaster-related additional information of the SMT, and the disaster-related additional information received through the file is displayed on a portion of the screen. The disaster-related additional information may be various types of information such as image, video, and text which are related to the disaster information contained in the emergency alert message.

**[00729]** FIG. 60 illustrates a block diagram of a receiving system (i.e., broadcast receiver) to receive and process services such as a mobile service, an EAS service, and an NRT service and so on transmitted through mobile broadcast according to an embodiment of the present invention. Referring to FIG. 60, the arrow shown in dotted line indicates a data path, and the arrow shown in solid line indicates a control signal path.

**[00730]** The receiving system of FIG. 60 may include a controller 2100, a tuner 2111, a demodulator 2112, an equalizer 2113, a known sequence detector (or known data detector) 2114, a block decoder 2115, a primary Reed-Solomon (RS) frame decoder 2116, a secondary RS frame decoder 2117, a signaling decoder 2118, and a baseband operation controller 2119. The receiving system according to the present invention may further include an FIC handler 2121, a service manager

2122, a service signaling handler 2123, and a first storage unit 2124. The receiving system according to the present invention may further include a primary RS frame buffer 2131, a secondary RS frame buffer 2132, and a transport packet (TS) handler 2133. The receiving system according to the present invention may further include an Internet Protocol (IP) datagram handler 2141, a descrambler 2142, an User Datagram Protocol (UDP) datagram handler 2143, a Real-time Transport Protocol/Real-time Transport Control Protocol (RTP/RTCP) datagram handler 2144, a Network Time Protocol (NTP) datagram handler 2145, a service protection stream handler 2146, a second storage unit 2147, an Asynchronous Layered Coding/Layered Coding Transport (ALC/LCT) stream handler 2148, an Extensible Mark-up Language (XML) parser 2150, and a Field Device Tool (FDT) handler 2151. The receiving system according to the present invention may further include an Audio/Video (A/V) decoder 2161, a file decoder 2162, a third storage unit 2163, a middle ware (M/W) engine 2164, and a Service Guide (SG) handler 2165. The receiving system according to the present invention may further include an Electronic Program Guide (EPG) manager 2171, an application manager 2172, and an User Interface (UI) manager 2173.

**[00731]**        Herein, for simplicity of the description of the present invention, the tuner 2111, the demodulator 2112, the equalizer 2113, the known sequence detector (or known data detector) 2114, the block decoder 2115, the primary RS frame decoder 2116, the secondary RS frame decoder 2117, the signaling decoder 2118, and the baseband operation controller 2119 will be collectively referred to as a baseband processor 2110. The FIC handler 2121, the service manager 2122, the service signaling handler 2123, and the first storage unit 2124 will be collectively referred to as a service multiplexer 2120. The primary RS frame buffer 2131, the secondary RS frame buffer 2132, and the TS handler 2133 will be collectively

referred to as an IP adaptation module 2130. The IP datagram handler 2141, the descrambler 2142, the UDP datagram handler 2143, the RTP/RTCP datagram handler 2144, the NTP datagram handler 2145, the service protection stream handler 2146, the second storage unit 2147, the ALC/LCT stream handler 2148, the XML parser 2150, and the FDT handler 2151 will be collectively referred to as a common IP module 2140. The A/V decoder 2161, the file decoder 2162, the third storage unit 2163, the M/W engine 2164, and the SG handler 2165 will be collectively referred to as an application module 2160.

**[00732]** The terms used in FIG. 60 are general terms that are currently being broadly used. However, according to the advent of new technology, terms deemed to be most appropriate by the applicant are also arbitrarily used in the present invention. The definition of such terms will be described clearly and in detail during the description of the corresponding portion of the present invention. Therefore, the terms used in the present invention should be understood by the significance lying within the terms and not merely by the term itself.

**[00733]** The baseband operation controller 2119 configured as shown in FIG. 60 controls the operation of each block included in the baseband processor 2110.

**[00734]** By tuning the receiving system to the frequency of a specific physical channel (or physical transmission channel (PTC)), the tuner 2111 performs a role enabling the receiving system to receive main service data, which correspond to broadcast signals for fixed broadcast receiving systems, and mobile service data, which correspond to broadcast signals for mobile broadcast receiving systems. The mobile service data may include an emergency alert message. The tuner 2111 down-converts the frequency of the tuned specific channel to an intermediate frequency (IF) and outputs the IF to the demodulator 2112 and the known sequence detector 2114. At this point, the tuner 2111 may receive main service data and mobile

service data that are real-time data and receive non-real time service data. Herein, the non-real time service data may include additional information (i.e., disaster related-additional information).

**[00735]** The demodulator 2112 performs self-gain control, carrier wave recovery, and timing recovery on the passband digital IF signal being inputted from the tuner 2111, so as to create a baseband signal. Then, the demodulator 2112 outputs the read baseband signal to the equalizer 2113 and the known sequence detector 2114. When performing carrier wave recovery and timing recovery, the demodulator 2112 may use the known data symbol sequence received from the known sequence detector 2114), so as to enhance the demodulating performance.

**[00736]** The equalizer 2113 compensates the channel distortion included in the demodulated signal, thereby outputting the processed signal to the block decoder 2115. The equalizer 2113 may enhance the equalizing performance by using the known data symbol sequence received from the known sequence detector 2114. Also, the equalizer 2113 may receive feedback on the decoding result of the block decoder 2113, thereby enhancing the equalizing performance.

**[00737]** The known sequence detector 2114 detects the position of the known data being inputted by the transmitting system from the input/output data of the demodulator 2112, i.e., data prior to being processed with the demodulation process or data being partially processed with the demodulation process. Then, along with the detected position information, the known sequence detector 2114 outputs the known data sequence generated from the detected position to the demodulator 2112, the equalizer 2113, and the baseband operation controller 2119. Additionally, the known sequence detector 2114 provides to the block decoder 2115 with information enabling the block decoder 2115 to differentiate mobile service data processed with additional encoding by the

transmitting system from main service data that are not processed with any additional encoding.

**[00738]** If the data channel-equalized by the equalizer 2113 and inputted to the block decoder 2115 correspond to data processed with both block-encoding of serial concatenated convolution code (SCCC) method and trellis-encoding by the transmitting system (*i.e.*, data within the RS frame, signaling data), the block decoder 2115 may perform trellis-decoding and block-decoding as inverse processes of the transmitting system. On the other hand, if the data channel-equalized by the equalizer 2113 and inputted to the block decoder 2115 correspond to data processed only with trellis-encoding and not block-encoding by the transmitting system (*i.e.*, main service data), the block decoder 2115 may perform only trellis-decoding.

**[00739]** The signaling decoder 2118 performs decoding on signaling data inputted after being channel-equalized by the equalizer 2113. It is assumed that the signaling data (or signaling information) inputted to the signaling decoder 2118 correspond to data processed with both block-encoding and trellis-encoding by the transmitting system. According to an embodiment of the present invention, the signaling data includes TPC (Transmission Parameter Channel) data and FIC (Fast Information Channel). For example, the signaling decoder 2118 performs regressive turbo decoding using a parallel concatenated convolution code (PCCC) method on the data of the signaling information region among the inputted data. Then, the signaling decoder 2118 separates the FIC data and TPC data from the turbo-decoded signaling data. Also, the signaling decoder 2118 performs RS-decoding on the separated TPC data as an inverse process of the transmitting system, thereby outputting the RS-decoded TPC data to the baseband operation controller 2119. Furthermore, the signaling decoder 2118 performs deinterleaving on the separated FIC data in subframe

units and then performs RS-decoding on the deinterleaved FIC data as an inverse process of the transmitting system, thereby outputting the RS-decoded data to the FIC handler 2121. The transmission unit of the FIC data being deinterleaved and RS-decoded by the signaling decoder 2118 and being outputted to the FIC handler 2121 corresponds to FIC segments.

**[00740]** The FIC handler 2121 receives FIC data from the signaling decoder 2118, so as to extract signaling information for service acquisition (*i.e.*, mapping information between an ensemble and a mobile service). In order to do so, the FIC handler 2121 may include an FIC segment buffer, an FIC segment parser, and an FIC chunk parser.

**[00741]** The FIC segment buffer buffers FIC segment groups being inputted in M/H frame units from the signaling decoder 2118, thereby outputting the buffered FIC segments to the FIC segment parser. Thereafter, the FIC segment parser extracts the header of each FIC segment stored in the FIC segment buffer so as to analyze the extracted headers. Then, based upon the analyzed result, the FIC segment parser outputs the payload of the respective FIC segments to the FIC chunk parser. The FIC chunk parser uses the analyzed result outputted from the FIC segment parser so as to recover the FIC chunk data structure from the FIC segment payloads, thereby analyzing the received FIC chunk data structure. Subsequently, the FIC chunk parser extracts the signaling information for service acquisition. The signaling information acquired from the FIC chunk parser is outputted to the service manager 2122. According to an embodiment of the present invention, the FIC segment header includes a `wake_up_indicator` field. Reference may be made to FIG. 39 or FIG. 40 for the detailed description of the `wake_up_indicator` field. Furthermore, according to an embodiment of the present invention, the FIC chunk payload includes an `EAS_ensemble_indicator` field. Reference may be

made to FIG. 42 for the detailed description of the EAS\_ensemble\_indicator field.

**[00742]** Meanwhile, the service signaling handler 2123 is configured of a service signaling buffer and a service signaling parser, and the service signaling handler 2123 buffers table sections, for example, SMT sections and EAT sections of a service signaling channel being transmitted from the UDP datagram handler 2143, thereby analyzing and processing the buffered table sections. The SMT information and the EAT information processed by the service signaling handler 2123 is also outputted to the service manager 2122.

**[00743]** According to an embodiment of the present invention, the service signaling channel transmitting the SMT section and/or the EAT section is included in a corresponding RS frame in a UDP/IP packet form having well-known destination IP address and well-known destination UDP port number and is received. Accordingly, the receiving system may parse the SMT section and descriptors of the SMT section without requiring extra information. This is identically applied to the EAT.

**[00744]** Furthermore, the SMT section provides signaling information (including IP access information) about all services within the ensemble that the SMT section is included. Therefore, the receiving system may access an IP stream component belonging to a desired service by using information parsed from the SMT section, thereby providing the service to the user. In addition, the receiving system may access the emergency alert message by using EAS-related signaling information included in the EAT, thereby providing the disaster information to the user.

**[00745]** If the service is an NRT service, access information of an FLUTE session that transmits content/files configuring of the NRT service may be extracted from the SMT.

**[00746]** The information parsed from the SMT is collected by the service manager 2122 and is stored in the first storage

unit 2124. The service manager 2122 stores the information extracted from the SMT as a service map and a guide data form in the first storage unit 2124.

**[00747]** That is, service manager 2122 uses the signaling information collected (or gathered) from the FIC handler 2121 and the service signaling handler 2123 so as to configure a service map, and the service manager 2122 uses the service guide (SG) collected from the service guide (SG) handler 2165 so as to configure a program guide. Then, the service manager 2122 refers to the configured service map and program guide to control the baseband operation controller 2119 so that the user can receive the mobile service he (or she) wishes. Also, depending upon the user's input, the service manager 2122 may perform controlling operations enabling the program guide to be displayed on at least one portion of the display screen.

**[00748]** The first storage unit 2124 stores the service map and service guide drawn up by the service manager 2122. Also, based upon the requests from the service manager 2122 and the EPG manager 2171, the required data are extracted from the first storage unit 2124, which are then transferred to the service manager 2122 and/or the EPG manager 2171.

**[00749]** The baseband operation controller 2119 receives the known data place information and TPC data, thereby transferring M/H frame time information, information indicating whether or not a data group exists in a selected parade, place information of known data within a corresponding data group, power control information, and so on to each block within the baseband processor 2110.

**[00750]** Meanwhile, according to the present invention, the transmitting system uses RS frames by encoding units. Herein, the RS frame may be divided into a primary RS frame and a secondary RS frame. However, according to the embodiment of the present invention, the primary RS frame and the secondary

RS frame will be divided based upon the level of importance of the corresponding data.

**[00751]** The primary RS frame decoder 2116 receives the data outputted from the block decoder 2115. At this point, according to the embodiment of the present invention, the primary RS frame decoder 2116 receives the mobile service data or NRT service data that have been Reed-Solomon (RS)-encoded and/or cyclic redundancy check (CRC)-encoded from the block decoder 2115. The primary RS frame decoder 2116 receives the SMT section data or the OMA BCAST SG data that have been Reed-Solomon (RS)-encoded and/or cyclic redundancy check (CRC)-encoded from the block decoder 2115.

**[00752]** In other words, the primary RS frame decoder 2116 receives at least one of the mobile service data, the NRT service data, the SMT section data and the OMA BCAST SG data but no main service data. The primary RS frame decoder 2116 performs inverse processes of the RS frame encoder (not shown) included in the digital broadcast transmitting system, thereby correcting errors existing within the primary RS frame. More specifically, the primary RS frame decoder 2116 forms a primary RS frame by grouping a plurality of data groups and, then, correct errors in primary RS frame units. In other words, the primary RS frame decoder 2116 decodes primary RS frames, which are being transmitted for actual broadcast services. The primary RS frame decoded by the primary RS frame decoder 2116 is output to the primary RS frame buffer 2131. The primary RS frame buffer 2131 buffers the primary RS frame, and then configures an M/H TP in each row unit. The M/H TPs of the primary RS frame outputs to the TP handler 2133.

**[00753]** Additionally, the secondary RS frame decoder 2117 receives the data outputted from the block decoder 2115. At this point, according to the embodiment of the present invention, the secondary RS frame decoder 2117 receives the mobile service data or the NRT service data that have been RS-

encoded and/or CRC-encoded from the block decoder 2115. The secondary RS frame decoder 2117 receives the SMT section data or the OMA BCAST SG data that have been Reed-Solomon (RS)-encoded and/or cyclic redundancy check (CRC)-encoded from the block decoder 2115.

**[00754]** That is, the secondary RS frame decoder 2117 receives at least one of the mobile service data, the NRT service data, the SMT section data and the OMA BCAST SG data but no main service data. The secondary RS frame decoder 2117 performs inverse processes of the RS frame encoder (not shown) included in the digital broadcast transmitting system, thereby correcting errors existing within the secondary RS frame. More specifically, the secondary RS frame decoder 2117 forms a secondary RS frame by grouping a plurality of data groups and, then, correct errors in secondary RS frame units. In other words, the secondary RS frame decoder 2117 decodes secondary RS frames, which are being transmitted for mobile audio service data, mobile video service data, guide data, and so on. The secondary RS frame decoded by the secondary RS frame decoder 2117 is output to the secondary RS frame buffer 2132. The secondary RS frame buffer 2132 buffers the secondary RS frame, and then configures an M/H TP in each row unit. The M/H TPs of the secondary RS frame outputs to the TP handler 2133.

**[00755]** The TP handler 2133 consists of a TP buffer and a TP parser. The TP handler 2133 buffers the M/H TPs inputted from the primary RS frame buffer 2131 and the secondary RS frame buffer 2132, and then extracts and analyzes each header of the buffered M/H TPs, thereby recovering IP datagram from each payload of the corresponding M/H TPs. The recovered IP datagram is outputted to the IP datagram handler 2141.

**[00756]** The IP datagram handler 2141 consists of an IP datagram buffer and an IP datagram parser. The IP datagram handler 2141 buffers the IP datagram delivered from the TP handler 2133, and then extracts and analyzes a header of the

buffered IP datagram, thereby recovering UDP datagram from a payload of the corresponding IP datagram. The recovered UDP datagram is outputted to the UDP datagram handler 2143.

**[00757]** If the UDP datagram is scrambled, the scrambled UDP datagram is descrambled by the descrambler 2142, and the descrambled UDP datagram is outputted to the UDP datagram handler 2143. For example, when the UDP datagram among the received IP datagram is scrambled, the descrambler 2142 descrambles the UDP datagram by inputting an encryption key and so on from the service protection stream handler 2146, and outputs the descrambled UDP datagram to the UDP datagram handler 2143.

**[00758]** The UDP datagram handler 2143 consists of an UDP datagram buffer and an UDP datagram parser. The UDP datagram handler 2143 buffers the UDP datagram delivered from the IP datagram handler 2141 or the descrambler 2142, and then extracts and analyzes a header of the buffered UDP datagram, thereby recovering data transmitted through a payload of the corresponding UDP datagram. If the recovered data is an RTP/RTCP datagram, the recovered data is outputted to the RTP/RTCP datagram handler 2144. If the recovered data is also an NTP datagram, the recovered data is outputted to the NTP handler 2145. Furthermore, if the recovered data is a service protection stream, the recovered data is outputted to the service protection stream handler 2146. And, if the recovered data is an ALC/LCT stream, the recovered data is outputted to the ALC/LCT steam handler 2148. Furthermore, if the recovered data are the SMT section data or the EAT section data, the recovered data are outputted to the service signaling section handler 2123.

**[00759]** At this point, since the service signaling channel transmitting the SMT section or the EAT section is an IP datagram having well-known destination IP address and well-known destination UDP port number, the IP datagram handler

2141 and the UDP datagram handler 2143 may output the data included in the SMT section and/or the EAT section without requiring extra information to the service signaling section handler 2123.

**[00760]** The RTP/RTCP datagram handler 2144 consists of an RTP/RTCP datagram buffer and an RTP/RTCP datagram parser. The RTP/RTCP datagram handler 2144 buffers the data of RTP/RTCP structure outputted from the UDP datagram handler 2143, and then extracts A/V stream from the buffered data, thereby outputting the extracted A/V stream to the A/V decoder 2161. The A/V decoder 2161 decodes the audio and video streams outputted from the RTP/RTCP datagram handler 2144 using audio and video decoding algorithms, respectively. The decoded audio and video data is outputted to the presentation manager 2170. Herein, at least one of an AC-3 decoding algorithm, an MPEG 2 audio decoding algorithm, an MPEG 4 audio decoding algorithm, an AAC decoding algorithm, an AAC+ decoding algorithm, an HE AAC decoding algorithm, an AAC SBR decoding algorithm, an MPEG surround decoding algorithm, and a BSAC decoding algorithm can be used as the audio decoding algorithm and at least one of an MPEG 2 video decoding algorithm, an MPEG 4 video decoding algorithm, an H.264 decoding algorithm, an SVC decoding algorithm, and a VC-1 decoding algorithm can be used as the video decoding algorithm.

**[00761]** The NTP datagram handler 2145 consists of an NTP datagram buffer and an NTP datagram parser. The NTP datagram handler 2145 buffers data having an NTP structure, the data being outputted from the UDP datagram handler 2143. Then, the NTP datagram handler 2145 extracts an NTP stream from the buffered data. Thereafter, the extracted NTP stream is outputted to the A/V decoder 2161 so as to be decoded.

**[00762]** The service protection stream handler 2146 may further include a service protection stream buffer. Herein, the service protection stream handler 2146 buffers data

designated (or required) for service protection, the data being outputted from the UDP datagram handler 2143. Subsequently, the service protection stream handler 2146 extracts information required for descrambling from the extracted data. The information required for descrambling includes a key value, such as SKTM and LKTM. The information for descrambling is stored in the second storage unit 2147, and, when required, the information for descrambling is outputted to the descrambler 2142.

**[00763]** The ALC/LCT stream handler 2148 consists of an ALC/LCT stream buffer and an ALC/LCT stream parser. And, the ALC/LCT stream handler 2148 buffers data having an ALC/LCT structure, the data being outputted from the UDP datagram handler 2143. Then, the ALC/LCT stream handler 2148 analyzes a header and a header expansion of an ALC/LCT session from the buffered data. Based upon the analysis result of the header and header expansion of the ALC/LCT session, when the data being transmitted to the ALC/LCT session correspond to an XML structure, the corresponding data are outputted to an XML parser 2150. Alternatively, when the data being transmitted to the ALC/LCT session correspond to a file structure, the corresponding data are outputted to a file decoder 2162.

**[00764]** At this point, when the data that are being transmitted to the ALC/LCT session are compressed, the compressed data are decompressed by a decompressor 2149, thereby being outputted to the XML parser 2150 or the file decoder 2162.

**[00765]** The XML parser 2150 analyses the XML data being transmitted through the ALC/LCT session. Then, when the analyzed data correspond to data designated to a file-based service, the XML parser 2150 outputs the corresponding data to the FDT handler 2151. On the other hand, if the analyzed data correspond to data designated to a service guide, the XML

parser 2150 outputs the corresponding data to the SG handler 2165.

**[00766]** The FDT handler 2151 analyzes and processes a file description table of a FLUTE protocol, which is transmitted in an XML structure through the ALC/LCT session. The FDT handler 2151 is controlled by the service manager 2122 when the received file is the file for the NRT service. The file may include the disaster-related additional information.

**[00767]** The SG handler 2165 collects and analyzes the data designated for a service guide, the data being transmitted in an XML structure, thereby outputting the analyzed data to the service manager 2122.

**[00768]** The file decoder 2162 decodes the data having a file structure and being transmitted through the ALC/LCT session, thereby outputting the decoded data to the middleware engine 2164 or storing the decoded data in a third storage unit 2163. According to an embodiment of the present invention, the data of a file structure that are decoded by the file decoder 2162 include the NRT service data.

**[00769]** Herein, the middleware engine 2164 translates the file structure data (*i.e.*, the application) like the NRT service and executes the translated application. Thereafter, the application may be outputted to an output device, such as a display screen or speakers, through the application presentation manager 2170. According to an embodiment of the present invention, the middleware engine 2164 corresponds to a JAVA-based middleware engine.

**[00770]** Based upon a user-input, the EPG manager 2171 receives EPG data either through the service manager 2122 or through the SG handler 2165, so as to convert the received EPG data to a display format, thereby outputting the converted data to the presentation manager 2170.

**[00771]** The application manager 2172 performs overall management associated with the processing of application data

74420-663

(for example, NRT service data), which are being transmitted in object formats, file formats, and so on.

**[00772]** Furthermore, based upon a user-command inputted through the UI manager 2173, the operation controller 2100 controls at least one of the service manager 2122, the EPG manager 2171, the application manager 2172, and the presentation manager 2170, so as to enable the user-requested function to be executed.

**[00773]** The UI manager 2173 transfers the user-input to the operation controller 2100 through the UI.

**[00774]** Finally, the presentation manager 2170 provides at least one of the audio and video data being outputted from the A/V decoder 2161 and the EPG data being outputted from the EPG manager 2171 to the user through the speaker and/or display screen.

**[00775]** It will be apparent to those skilled in the art that various modifications and variations can be made in the present invention without departing from the scope of the inventions. Thus, it is intended that the present invention covers the modifications and variations of this invention provided they come within the scope of the appended claims and their equivalents.

**[Mode for Invention]**

**[00776]** As described above, the present invention is described with respect to the best mode for carrying out the present invention.

**[Industrial Applicability]**

**[00777]** As described above, the present invention may be fully (or entirely) or partially applied to broadcast and communication fields an embodiment of a transmitting system, a receiving system and method of processing a broadcast signal according to the present invention.

74420-663

CLAIM:

1. A method for processing a broadcast signal in a transmitting system, the method comprising:

performing Reed Solomon-Cyclic Redundancy Check (RS-  
5 CRC) encoding on mobile service data, thereby generating an RS frame belonging to an ensemble;

dividing the RS frame into a plurality of RS frame portions;

mapping the RS frame portions into each of data  
10 groups and inserting one fast information channel (FIC) segment and a plurality of known data sequences into each of the data groups;

performing trellis encoding on data of the data groups; and

15 transmitting the broadcast signal including the trellis-encoded data,

wherein an FIC chunk is configured with an FIC chunk header and FIC chunk payload, signals binding information between the ensemble and a mobile service included in the  
20 ensemble, and is divided into a plurality of FIC segment payloads, wherein the FIC segment is configured with an FIC segment header and one of the plurality of FIC segment payloads, wherein the FIC segment header comprises wake-up indication information which is used to determine at a receiver  
25 whether the receiver should wake up or not, and

74420-663

wherein an emergency alert message is transferred through at least an emergency alert table (EAT) or an IP datagram.

2. The method according to claim 1, wherein the emergency alert message is transmitted in a Common Alerting Protocol (CAP) form or a syntax form.

3. The method according to claim 1, wherein the ensemble comprises at least one of a service map table (SMT) describing the mobile service and the EAT, wherein each of the SMT and the EAT comprises an ensemble identifier to identify the ensemble, and wherein the EAT further includes information for indicating whether the emergency alert message is transferred through the EAT or the IP datagram.

4. The method according to claim 3, wherein IP datagrams including the SMT and IP datagrams including the EAT are transmitted over a service signaling channel, and wherein all the IP datagrams transmitted over the service signaling channel have the same well-known IP address and the same well-known UDP port number.

5. The method according to claim 4, wherein the SMT and the EAT are identified using a table identifier included in each of the tables.

6. The method according to claim 3, wherein the FIC chunk payload comprises the ensemble identifier to identify the ensemble and indication information to indicate whether the EAT is transmitted over a service signaling channel included in the ensemble.

74420-663

7. The method according to claim 3, wherein at least one of the SMT and the EAT further includes an non-real time (NRT) service identifier to identify an NRT service providing additional information associated with the emergency alert  
5 message.

8. The method according to claim 7, wherein the additional information comprises at least one of a text, an image and a video associated with the emergency alert message.

9. A transmitting system comprising:

10 a first encoder configured to perform Reed Solomon-Cyclic Redundancy Check (RS-CRC) encoding on mobile service data, thereby generating an RS frame belonging to an ensemble;

a divider configured to divide the RS frame into a plurality of RS frame portions;

15 a group formatter configured to map the RS frame portions into each of data groups and insert one fast information channel (FIC) segment and a plurality of known data sequences into each of the data groups;

a second encoder configured to perform trellis  
20 encoding on data of the data groups; and

a transmitting unit configured to transmit a broadcast signal including the trellis-encoded data,

wherein an FIC chunk is configured with an FIC chunk header and FIC chunk payload, signals binding information  
25 between the ensemble and a mobile service included in the ensemble, and is divided into a plurality of FIC segment

74420-663

payloads, wherein the FIC segment is configured with an FIC  
segment header and one of the plurality of FIC segment  
payloads, and wherein the FIC segment header comprises wake-up  
indication information which is used to determine at a receiver  
5 whether the receiver should wake up or not, and

wherein an emergency alert message is transferred  
through at least an emergency alert table (EAT) or an IP  
datagram.

10. The transmitting system according to claim 9, wherein  
10 the emergency alert message is transmitted in a Common Alerting  
Protocol (CAP) form or a syntax form.

11. The transmitting system according to claim 9, wherein  
the ensemble comprises at least one of a service map table  
(SMT) describing the mobile service and the EAT, wherein each  
15 of the SMT and the EAT comprises an ensemble identifier to  
identify the ensemble, and wherein the EAT further includes  
information for indicating whether the emergency alert message  
is transferred through the EAT or the IP datagram.

12. The transmitting system according to claim 11,  
20 wherein IP datagrams including the SMT and IP datagrams  
including the EAT are transmitted over a service signaling  
channel, and wherein all the IP datagrams transmitted over the  
service signaling channel have the same well-known IP address  
and the same well-known UDP port number.

25 13. The transmitting system according to claim 11,  
wherein the FIC chunk payload comprises the ensemble identifier  
to identify the ensemble and indication information to indicate

74420-663

whether the EAT is transmitted over a service signaling channel included in the ensemble.

14. The transmitting system according to claim 11, wherein at least one of the SMT and the EAT further includes an  
5 non-real time (NRT) service identifier to identify an NRT service providing additional information associated with the emergency alert message.

15. The transmitting system according to claim 14, wherein the additional information comprises at least one of a  
10 text, an image and a video associated with the emergency alert message.

FIG. 1

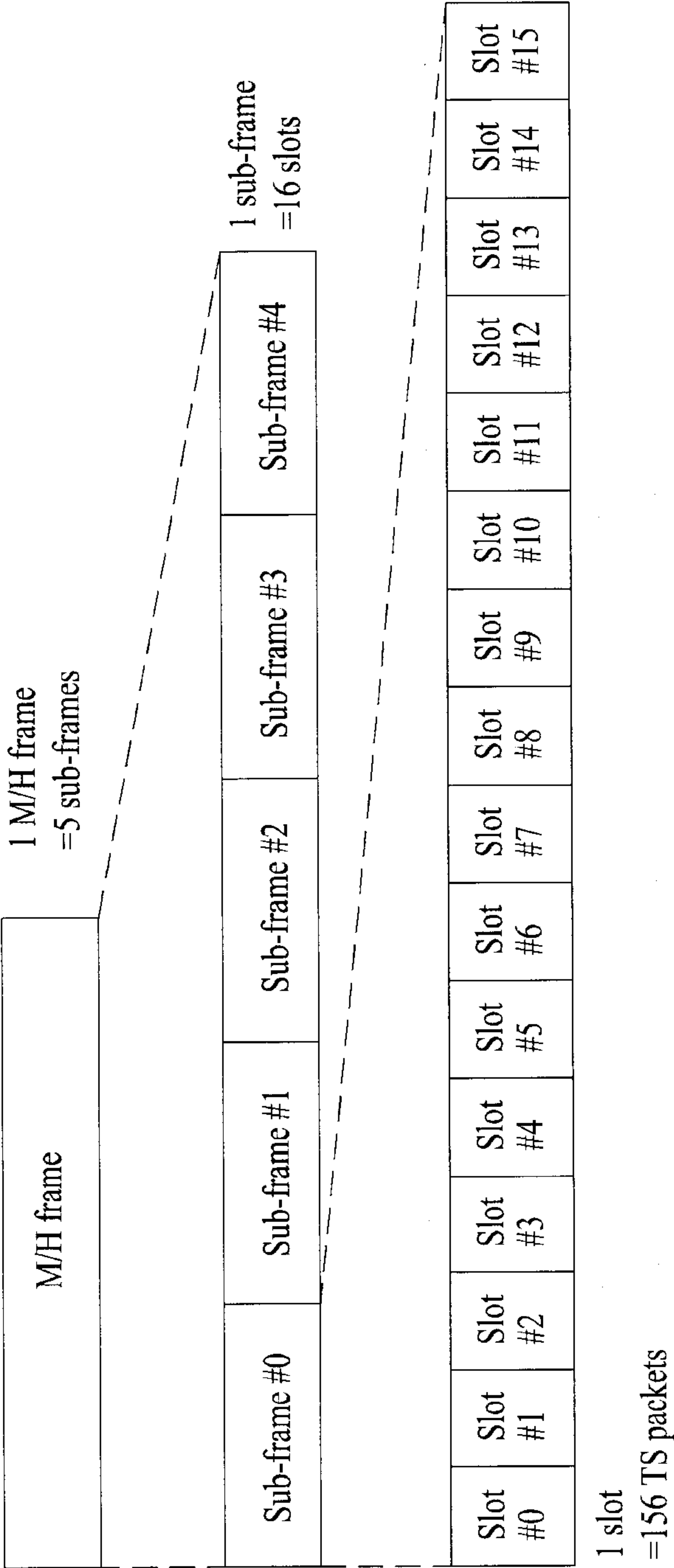


FIG. 2

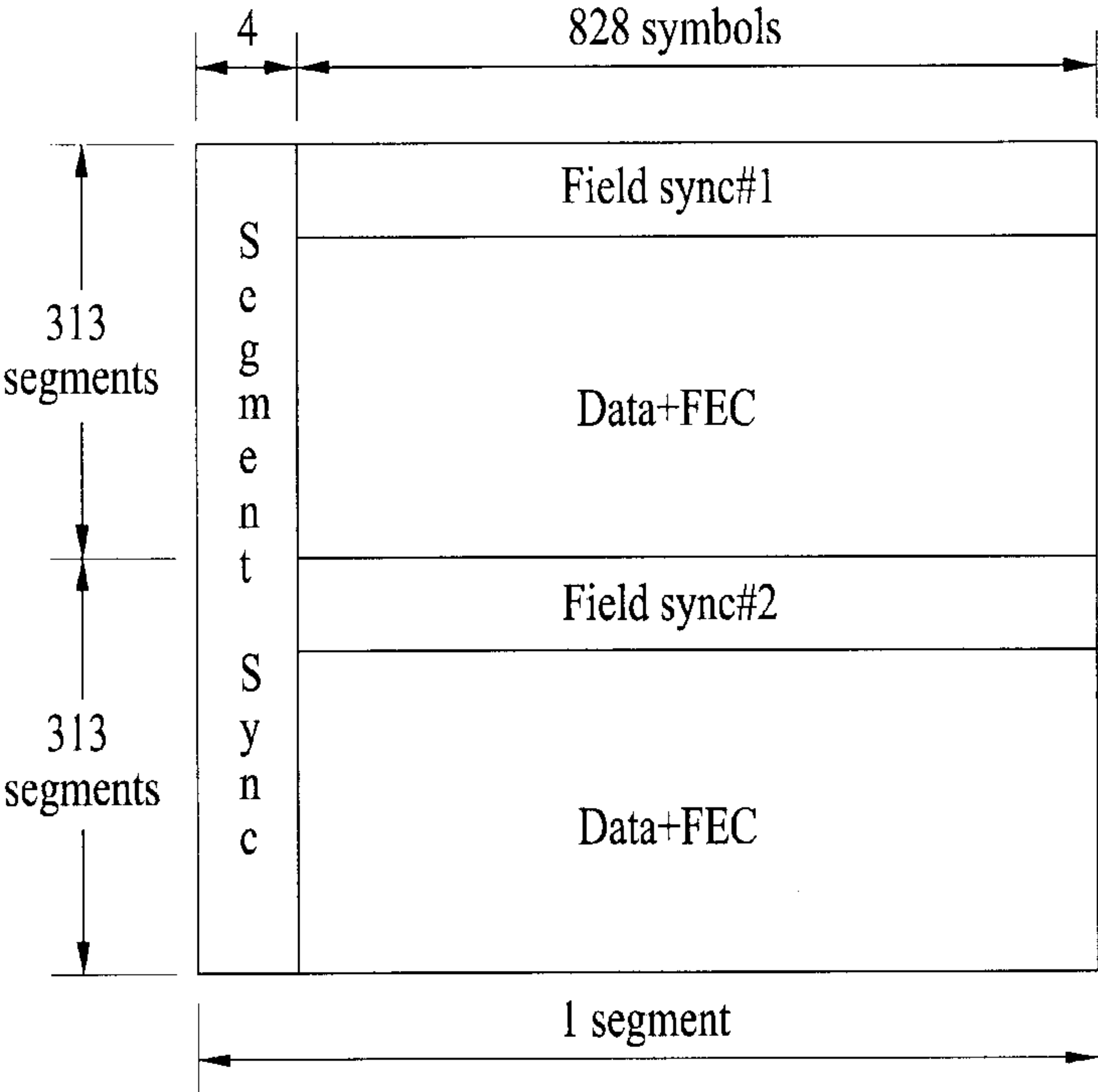


FIG. 3

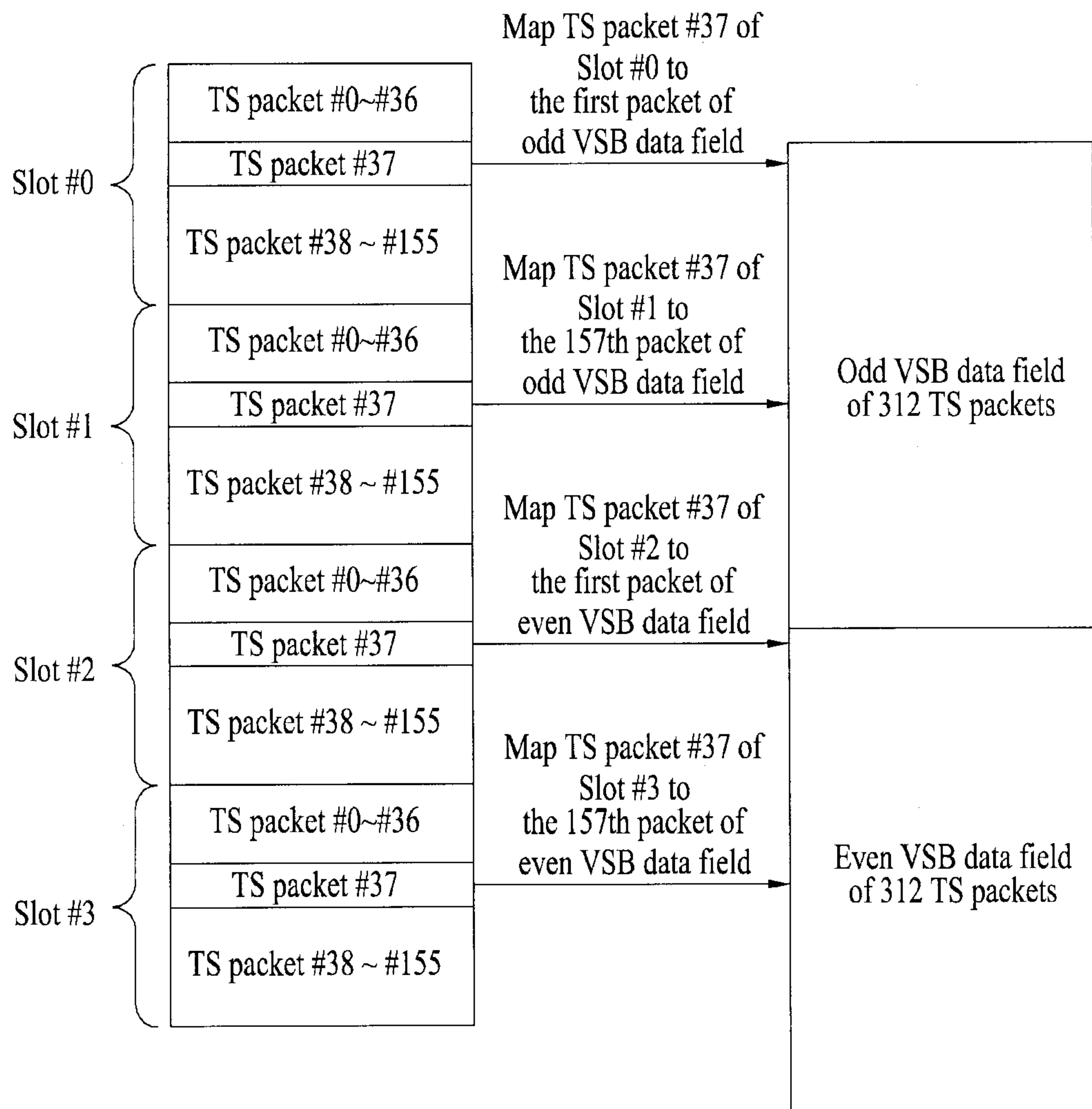


FIG. 4

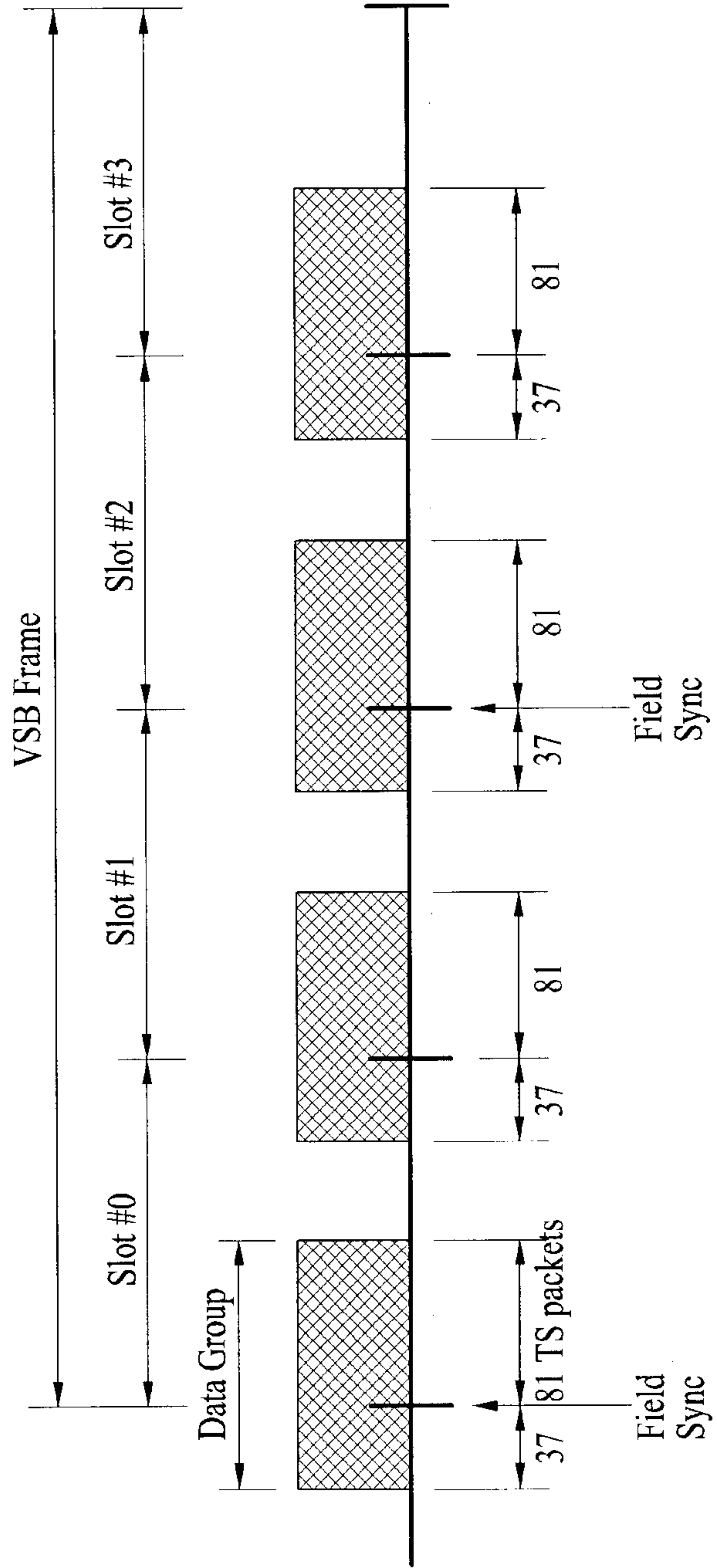


FIG. 5

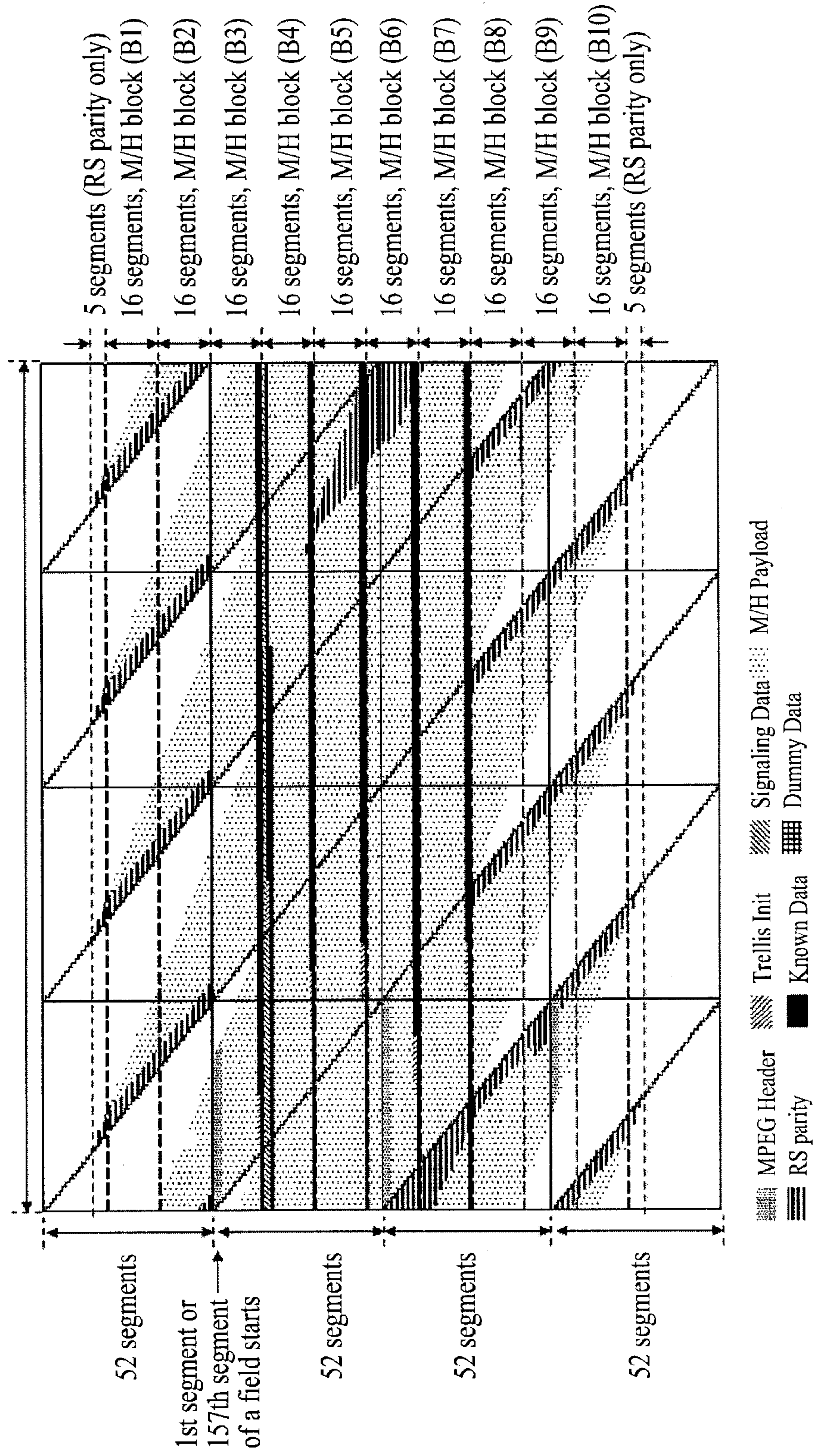


FIG. 6

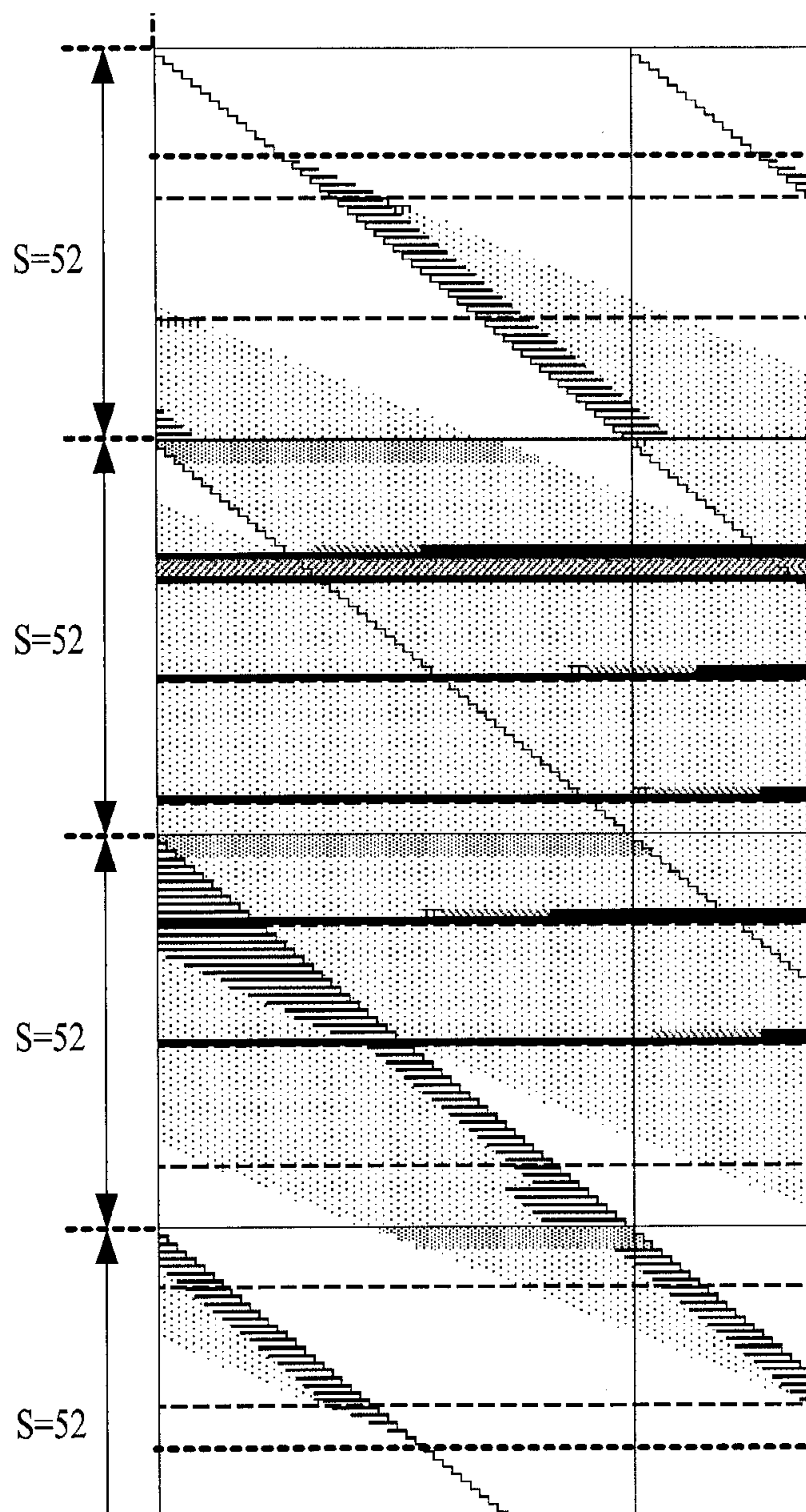


FIG. 7

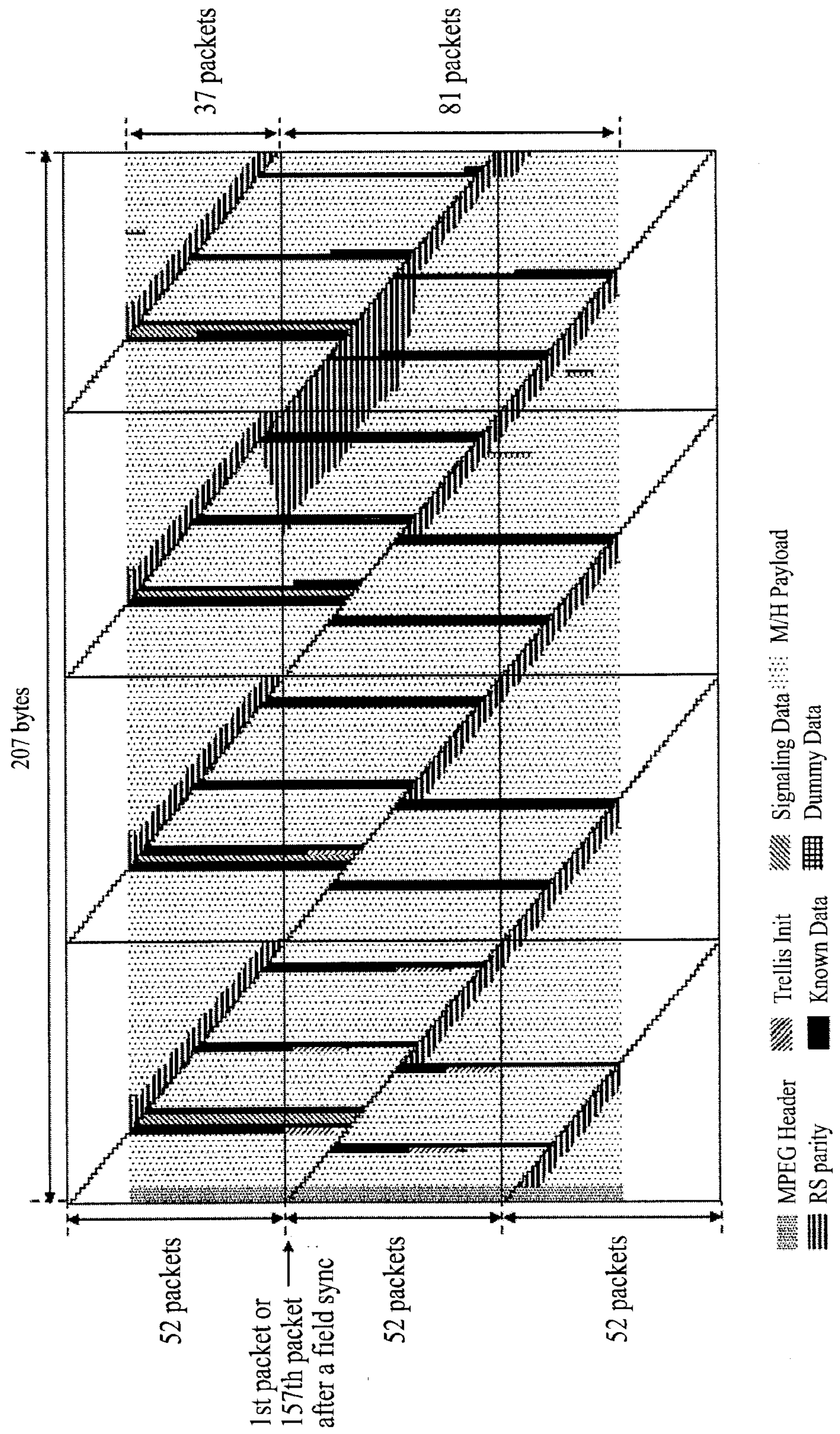


FIG. 8

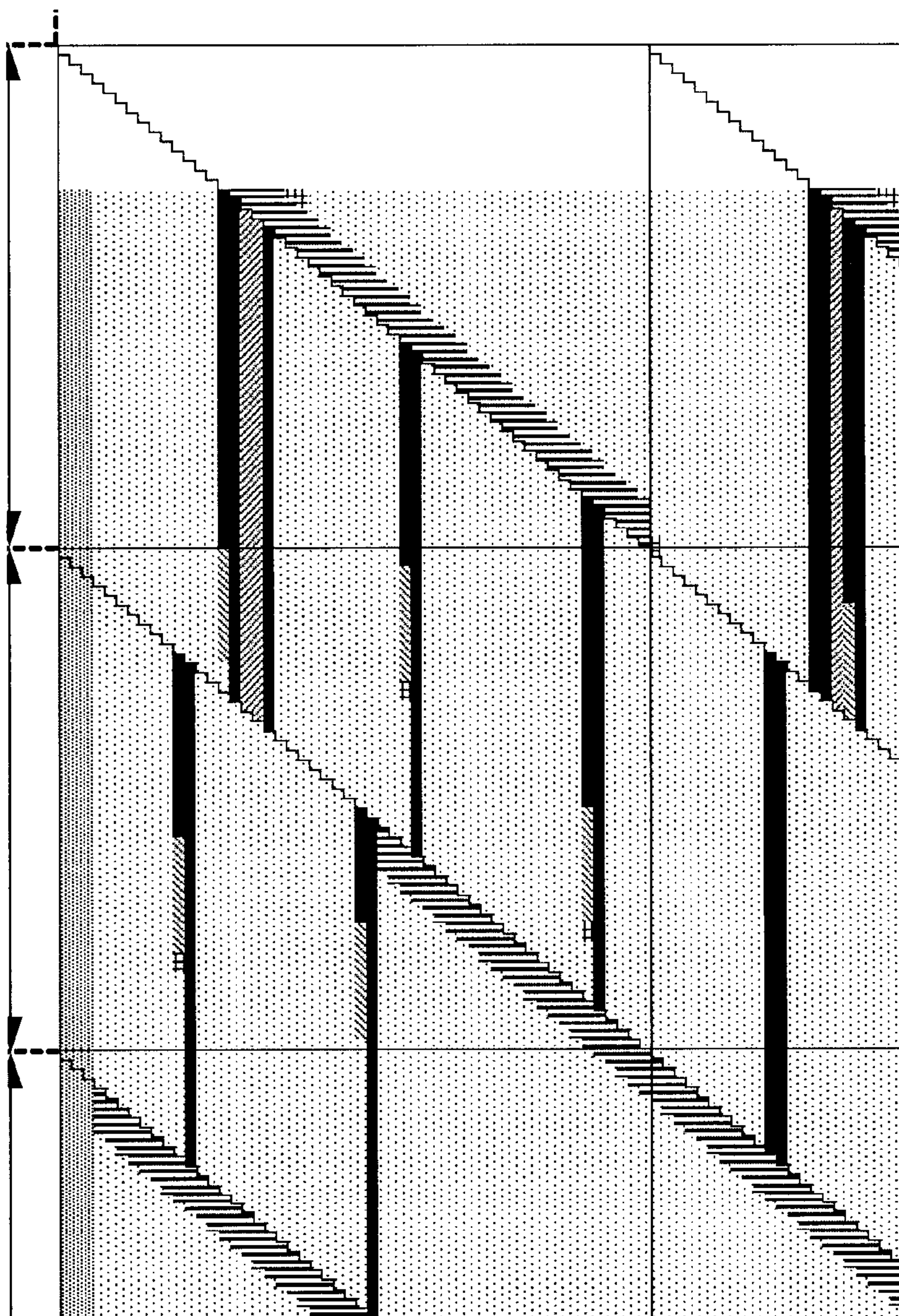


FIG. 9

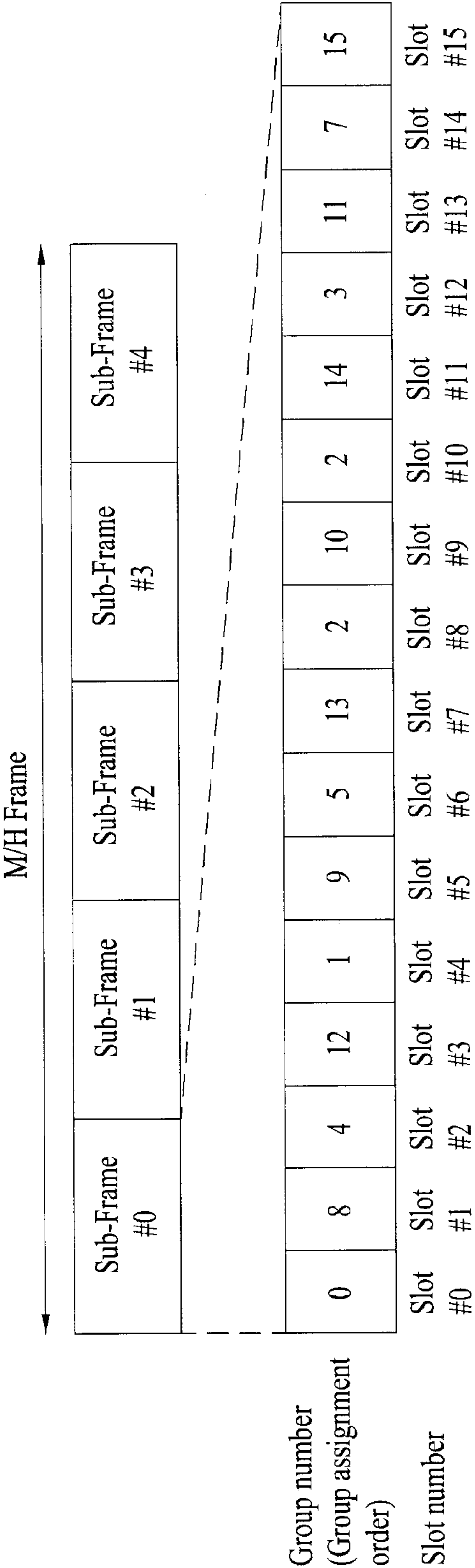


FIG. 10

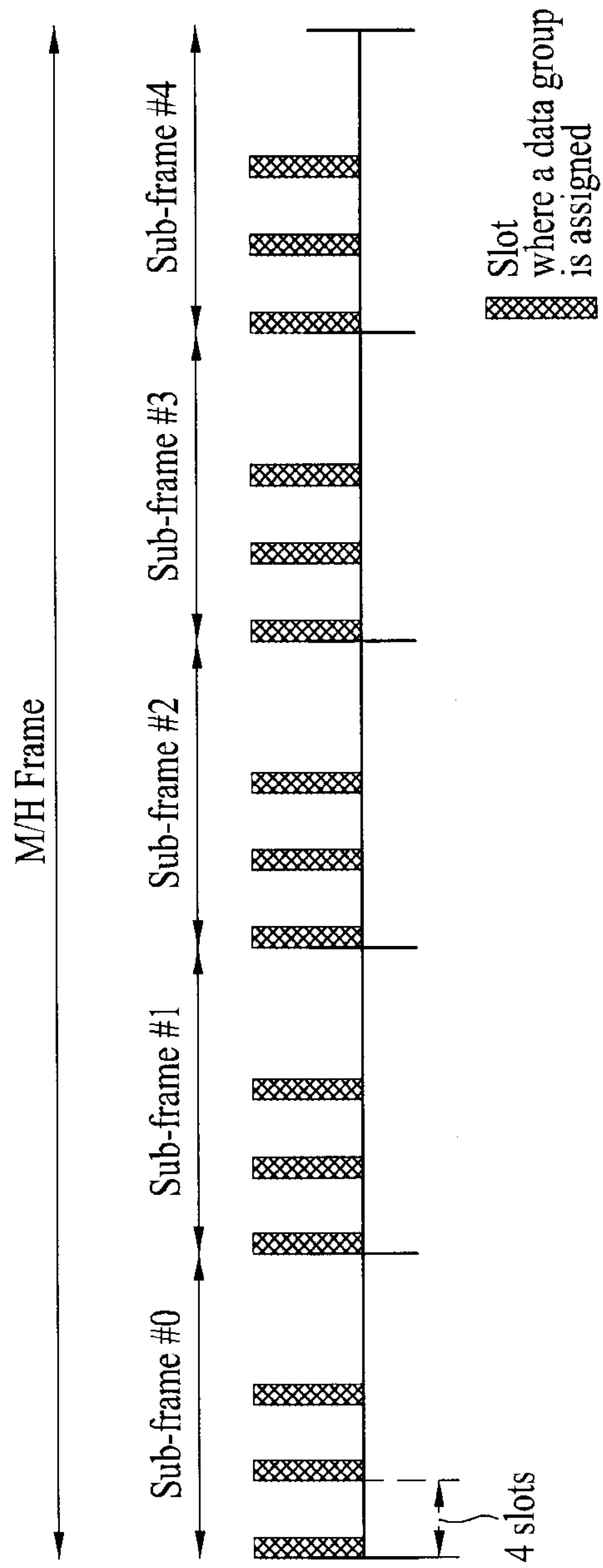


FIG. 11

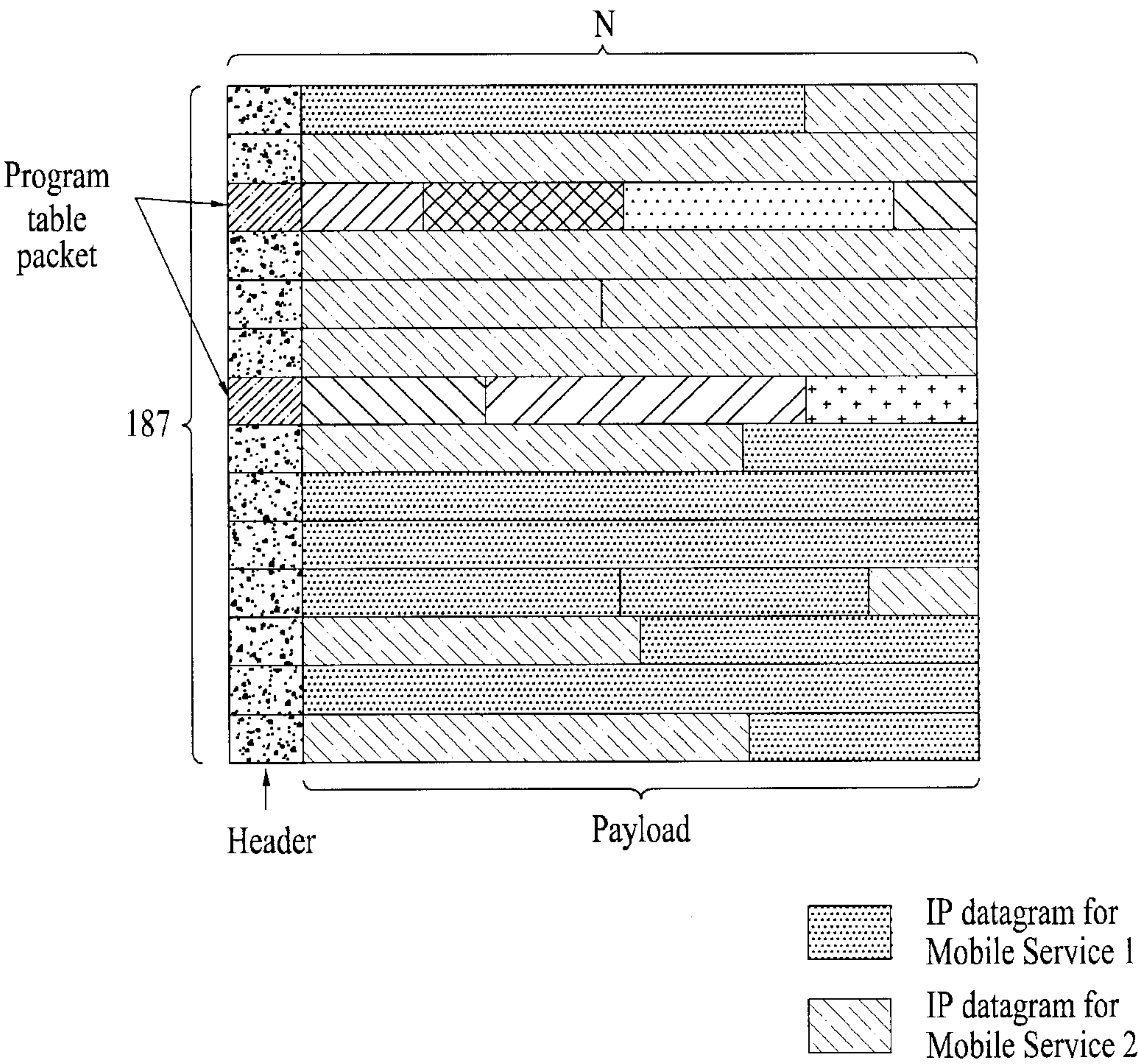


FIG. 12

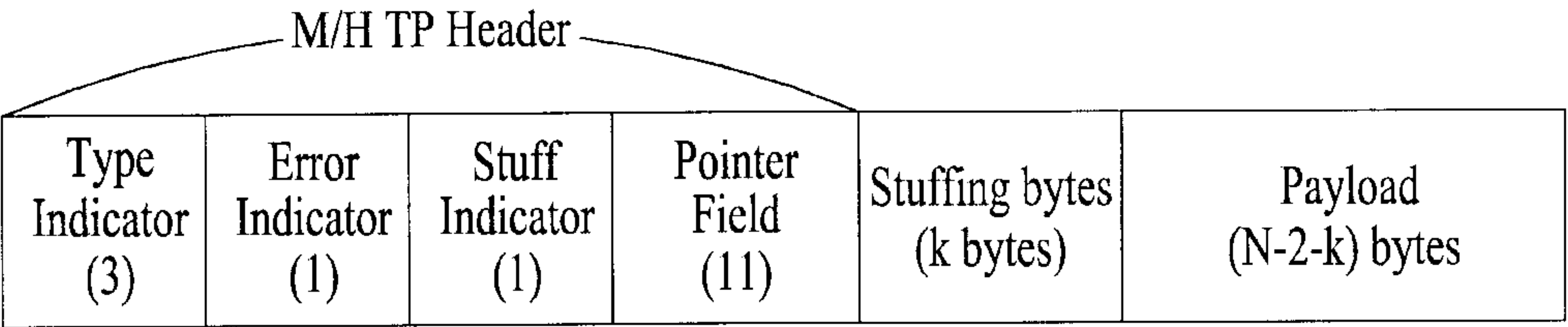


FIG. 13

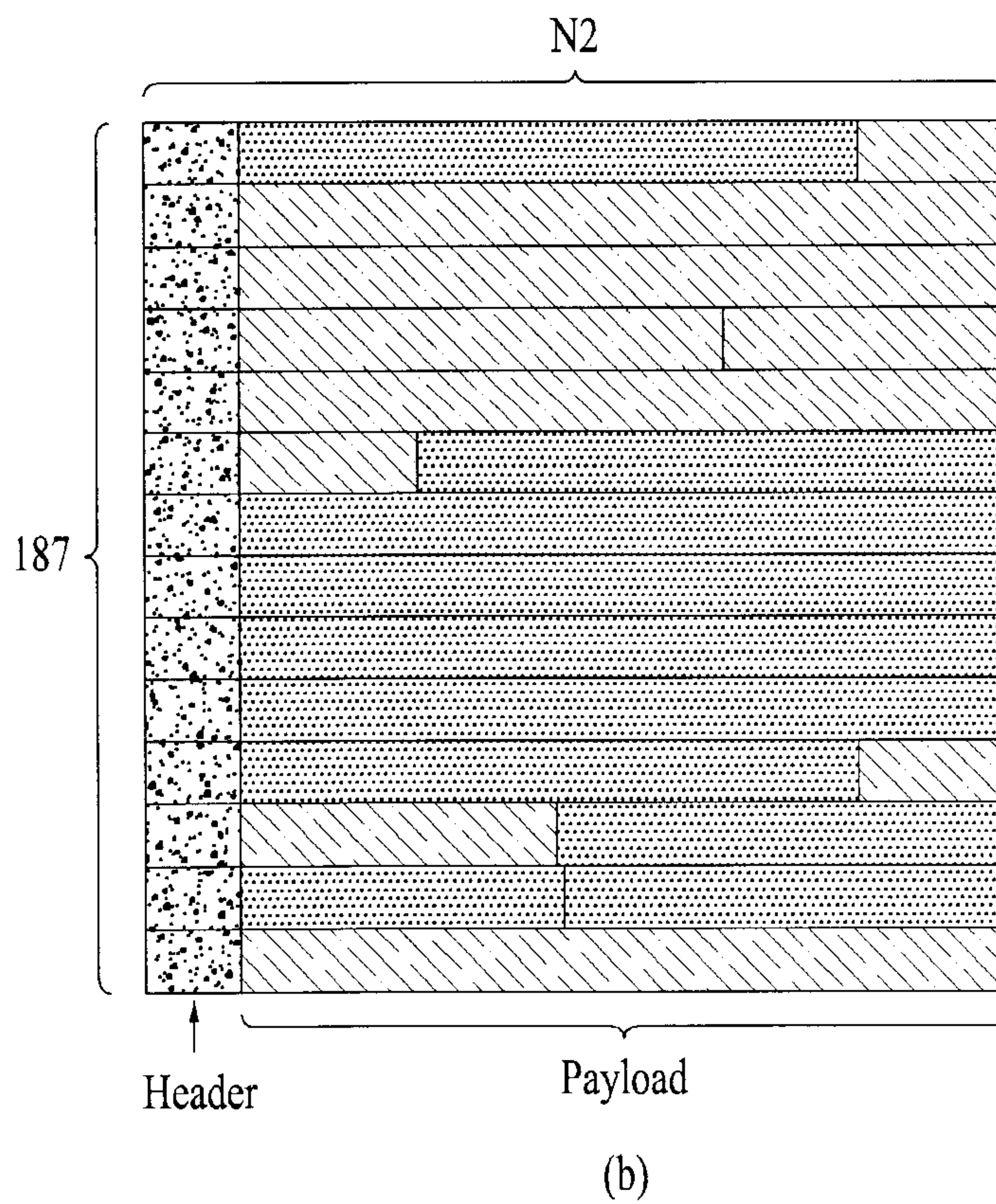
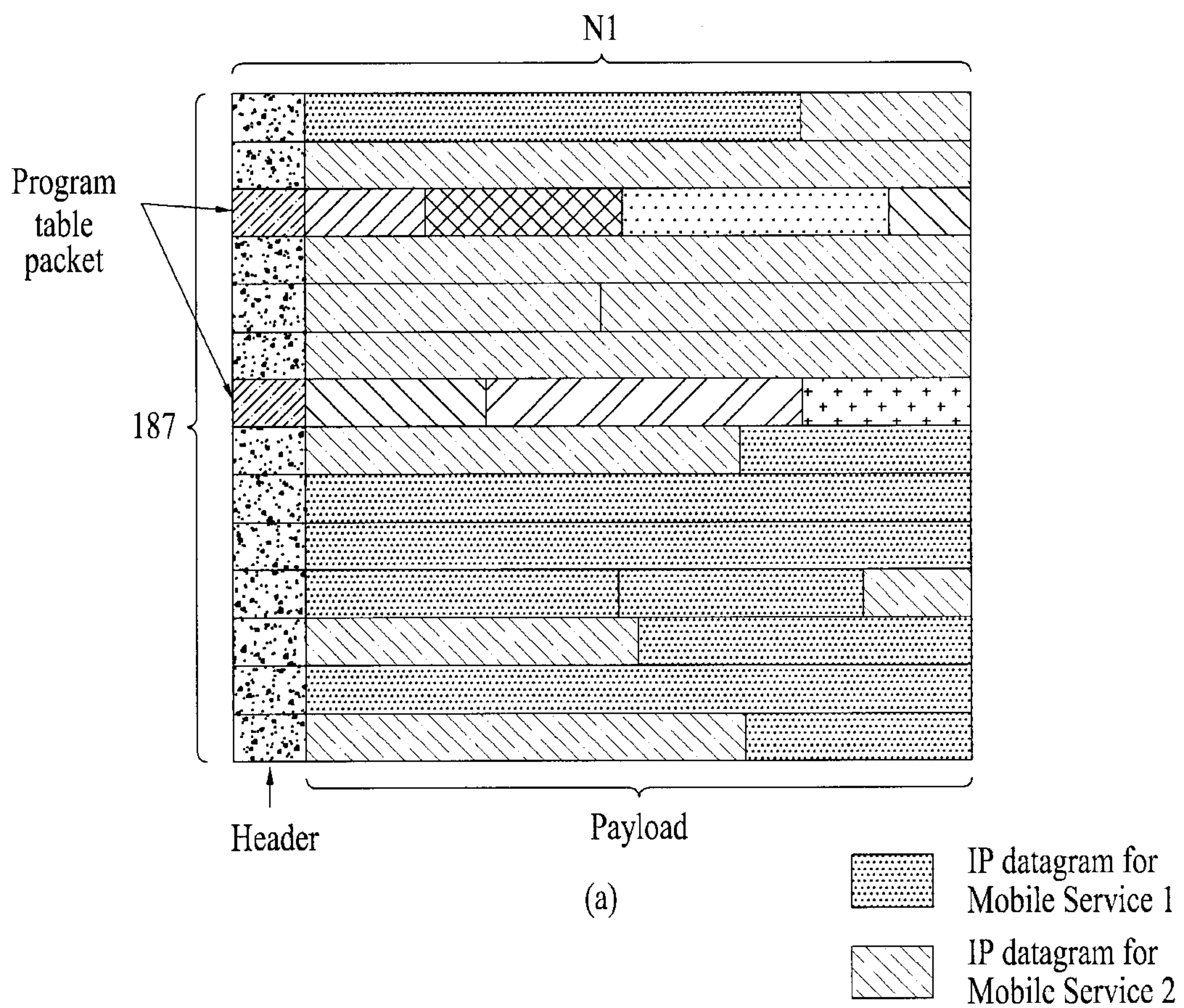


FIG. 14

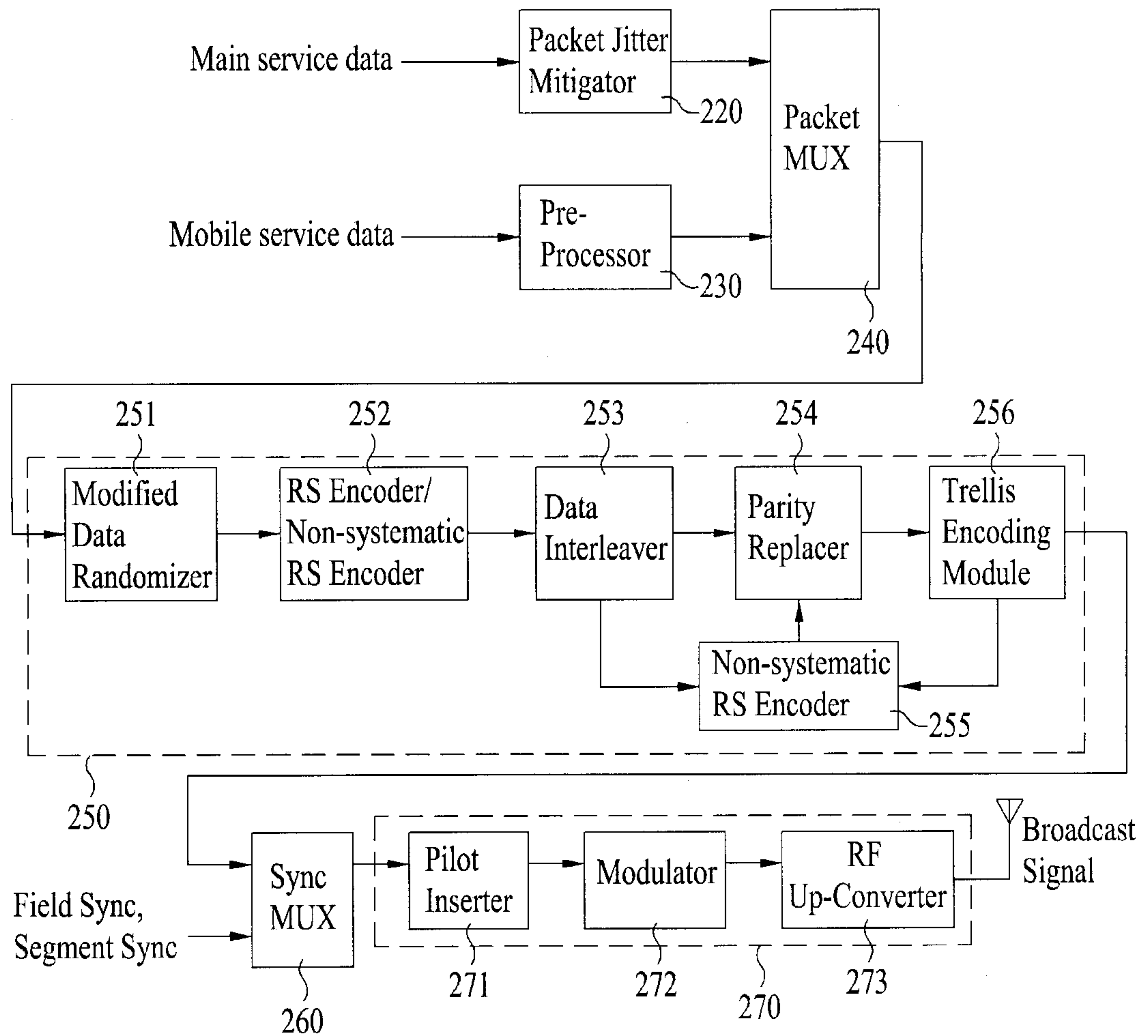


FIG. 15

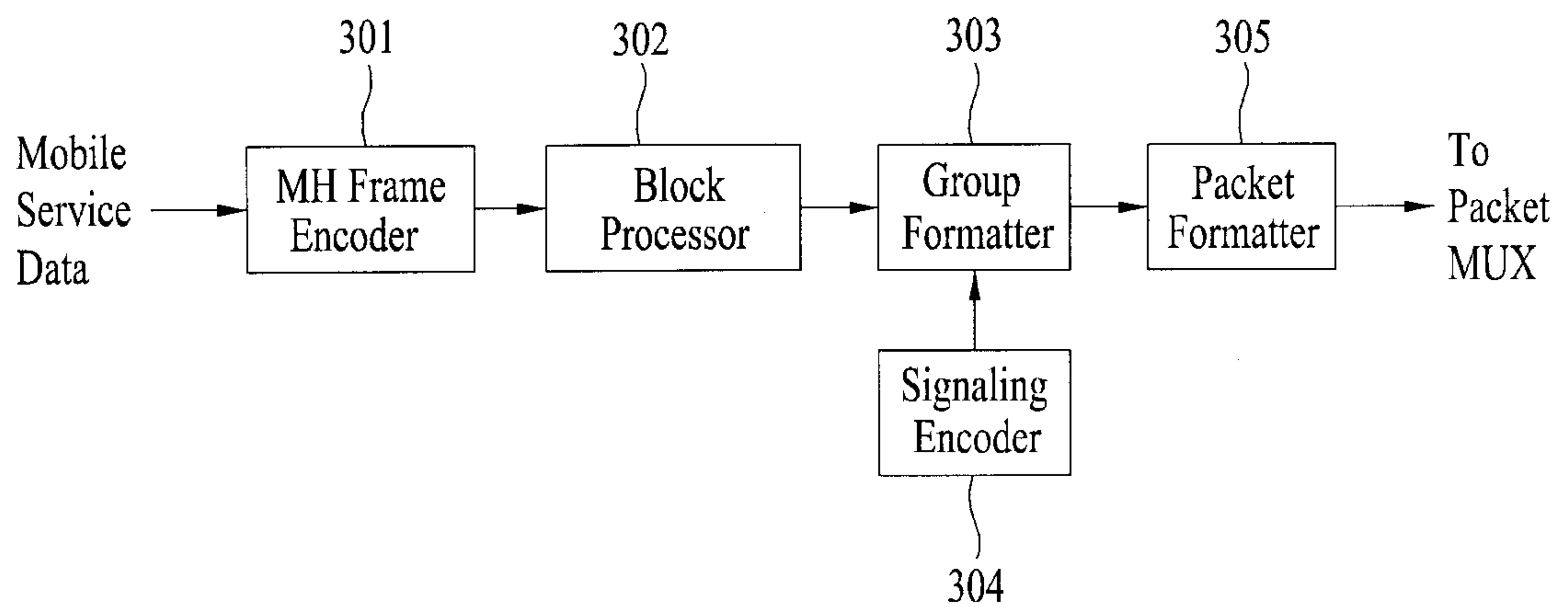


FIG. 16

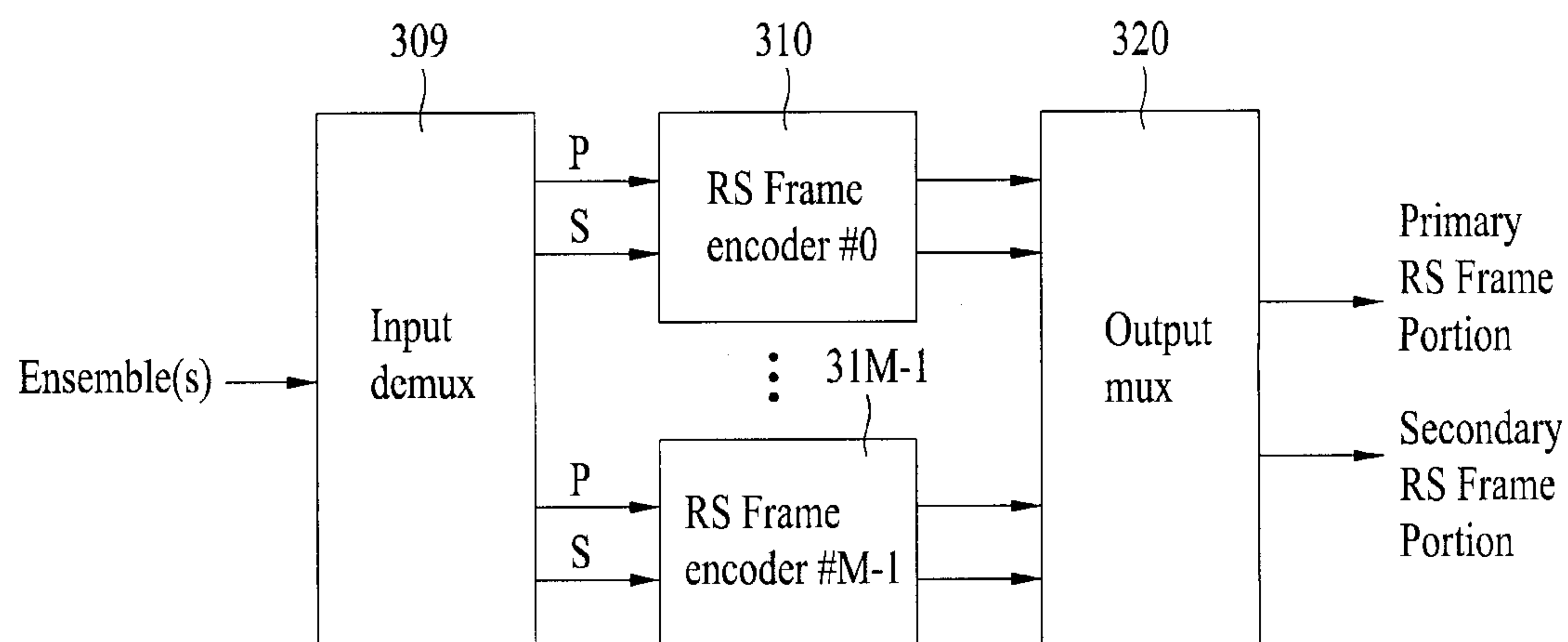


FIG. 17

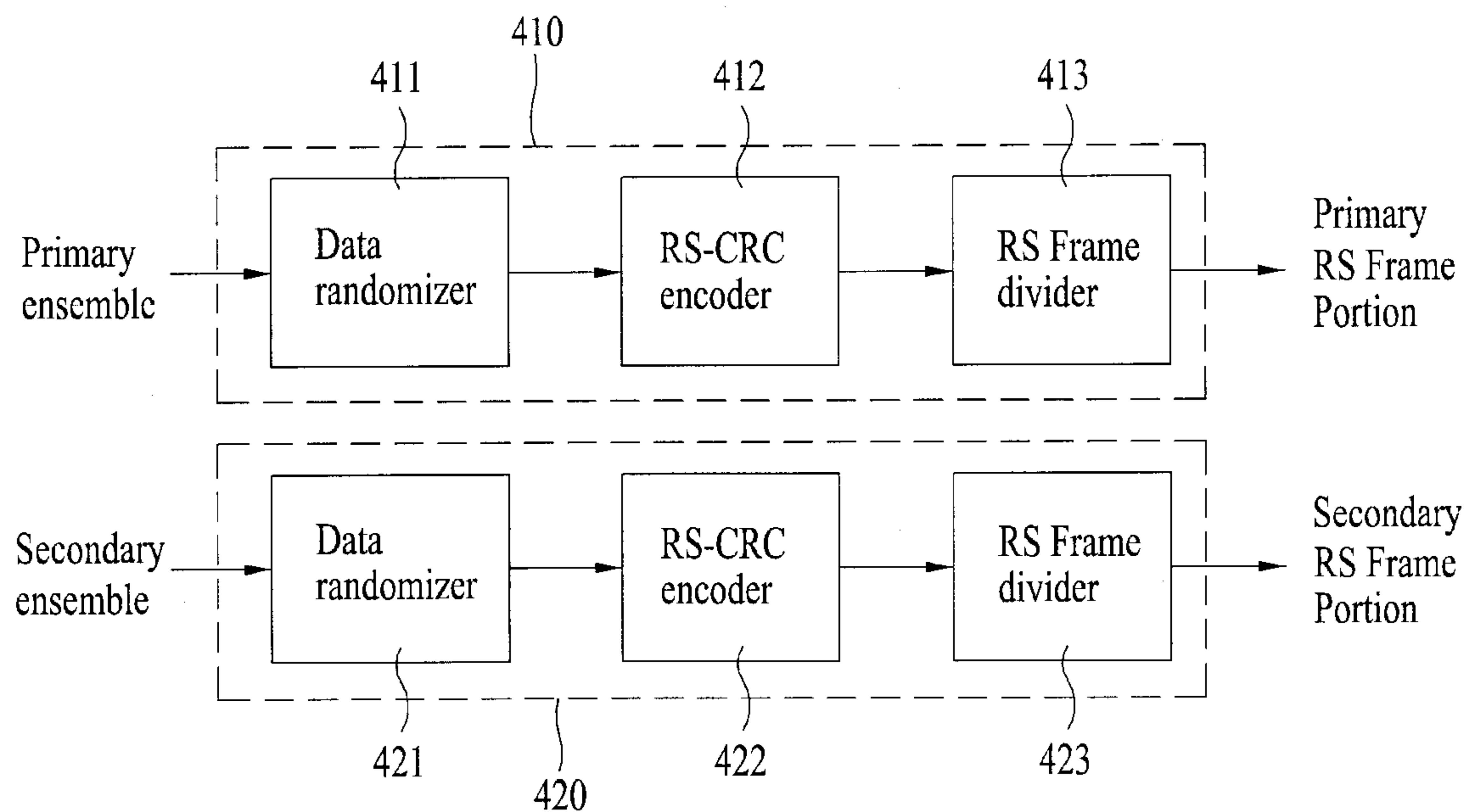


FIG. 18

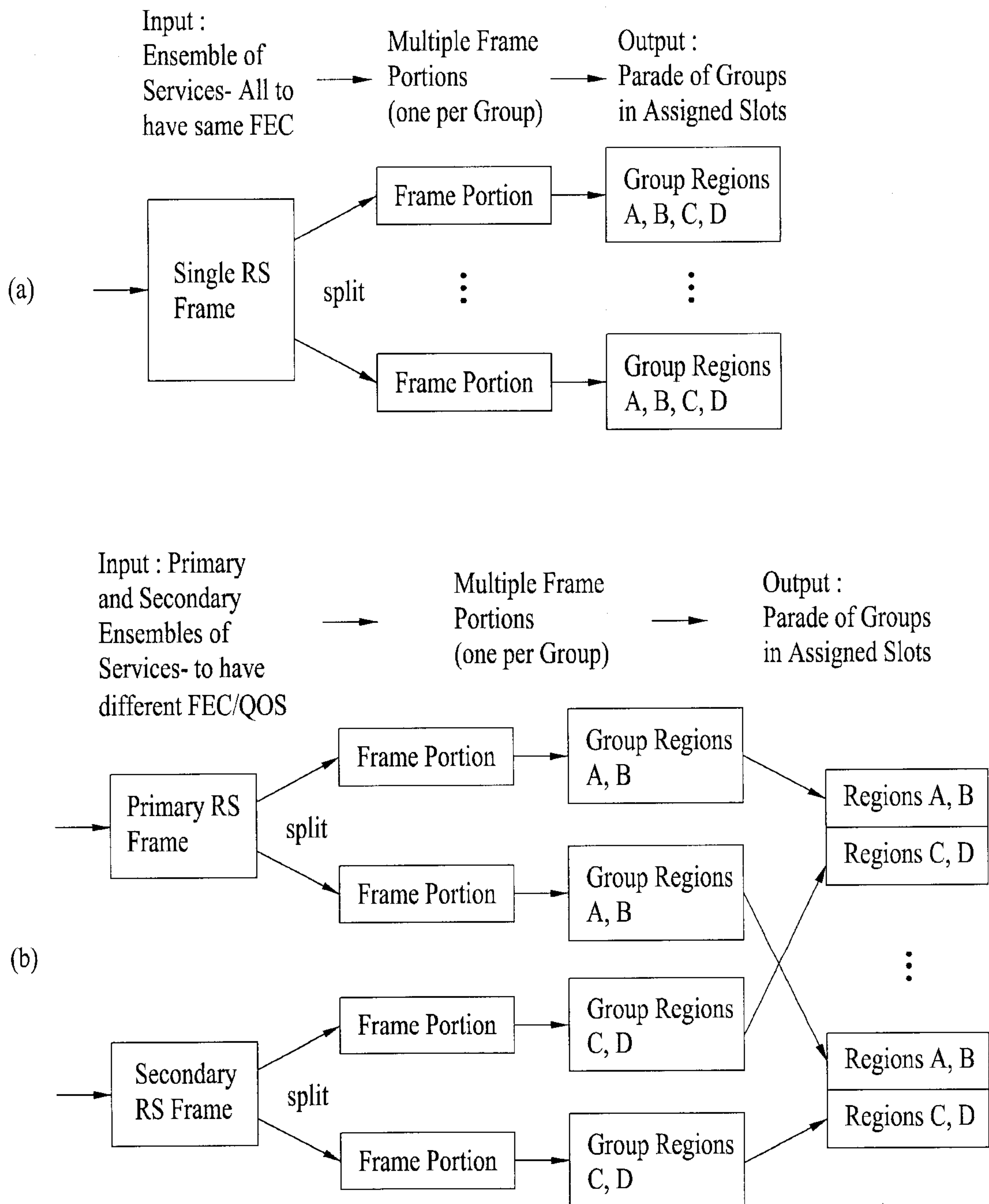


FIG. 19

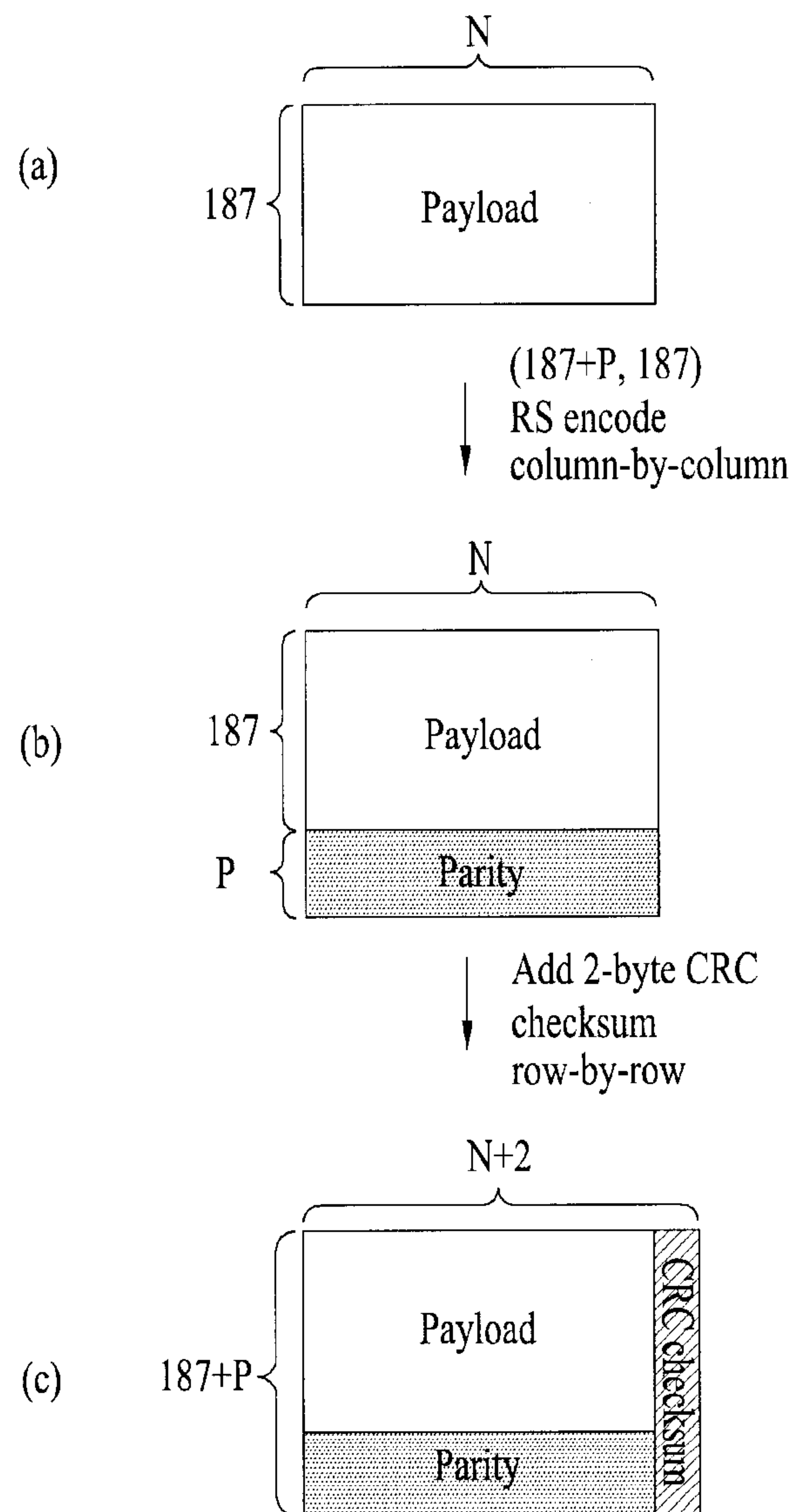


FIG. 20

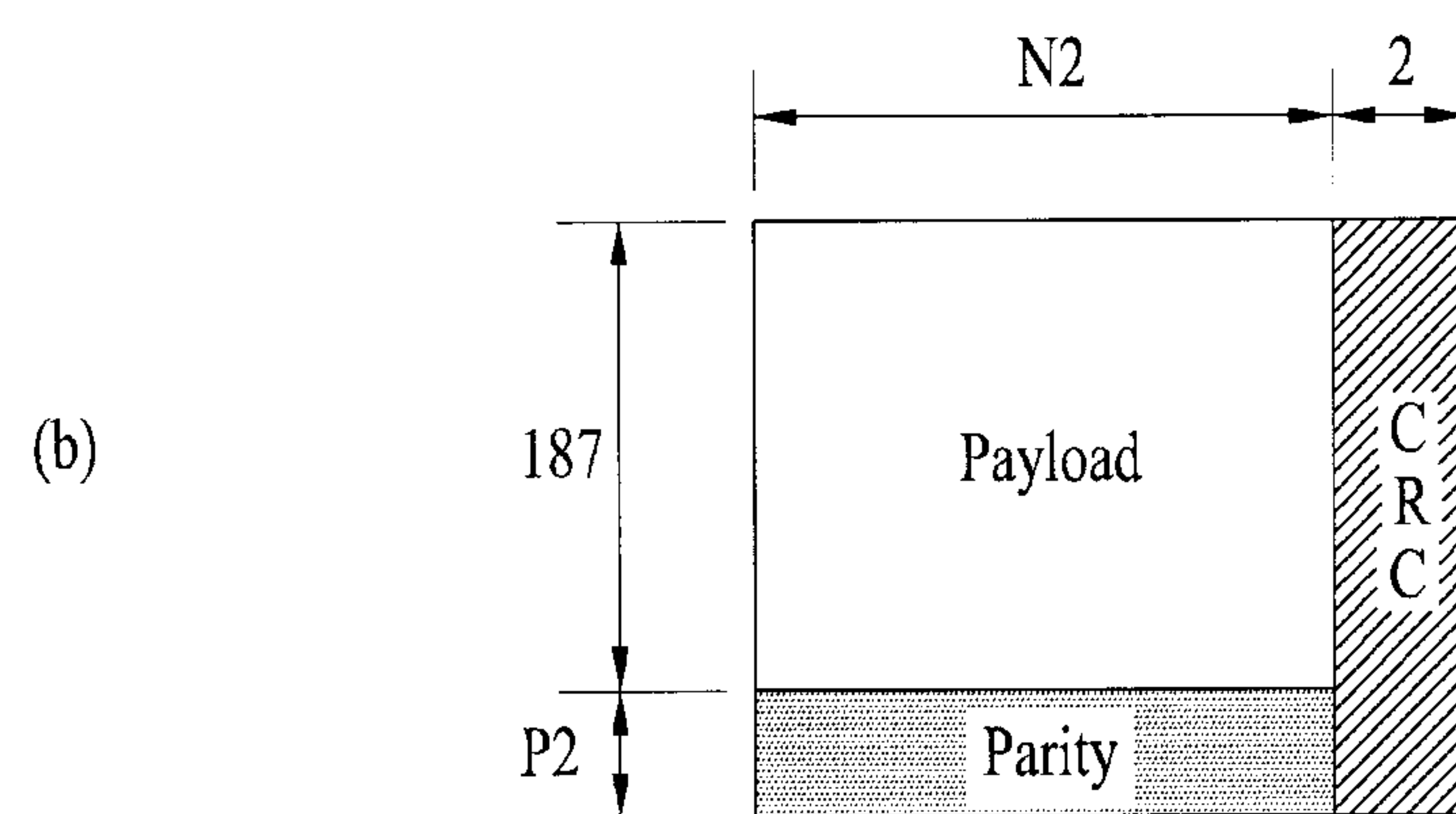
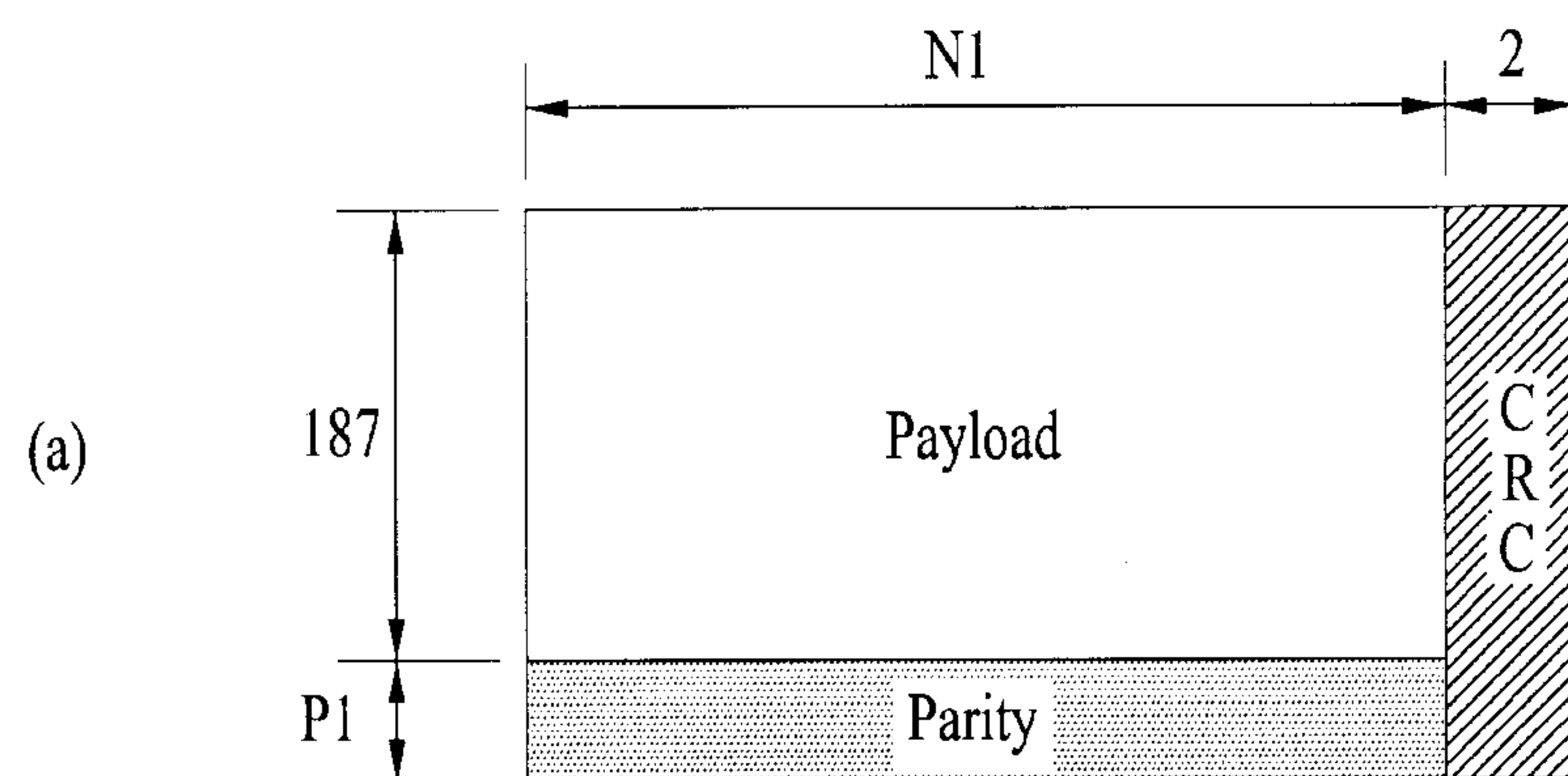


FIG. 21

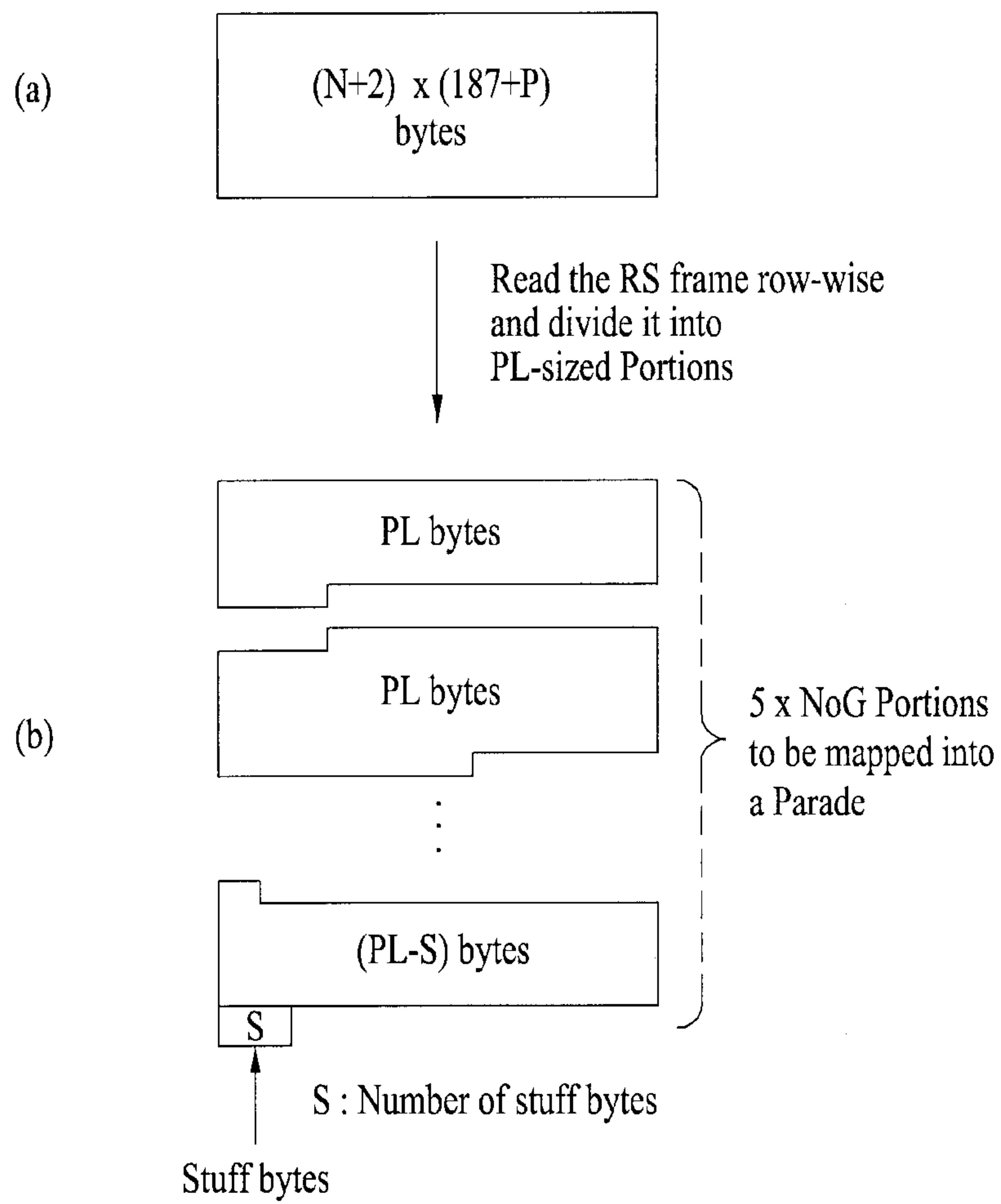


FIG. 22

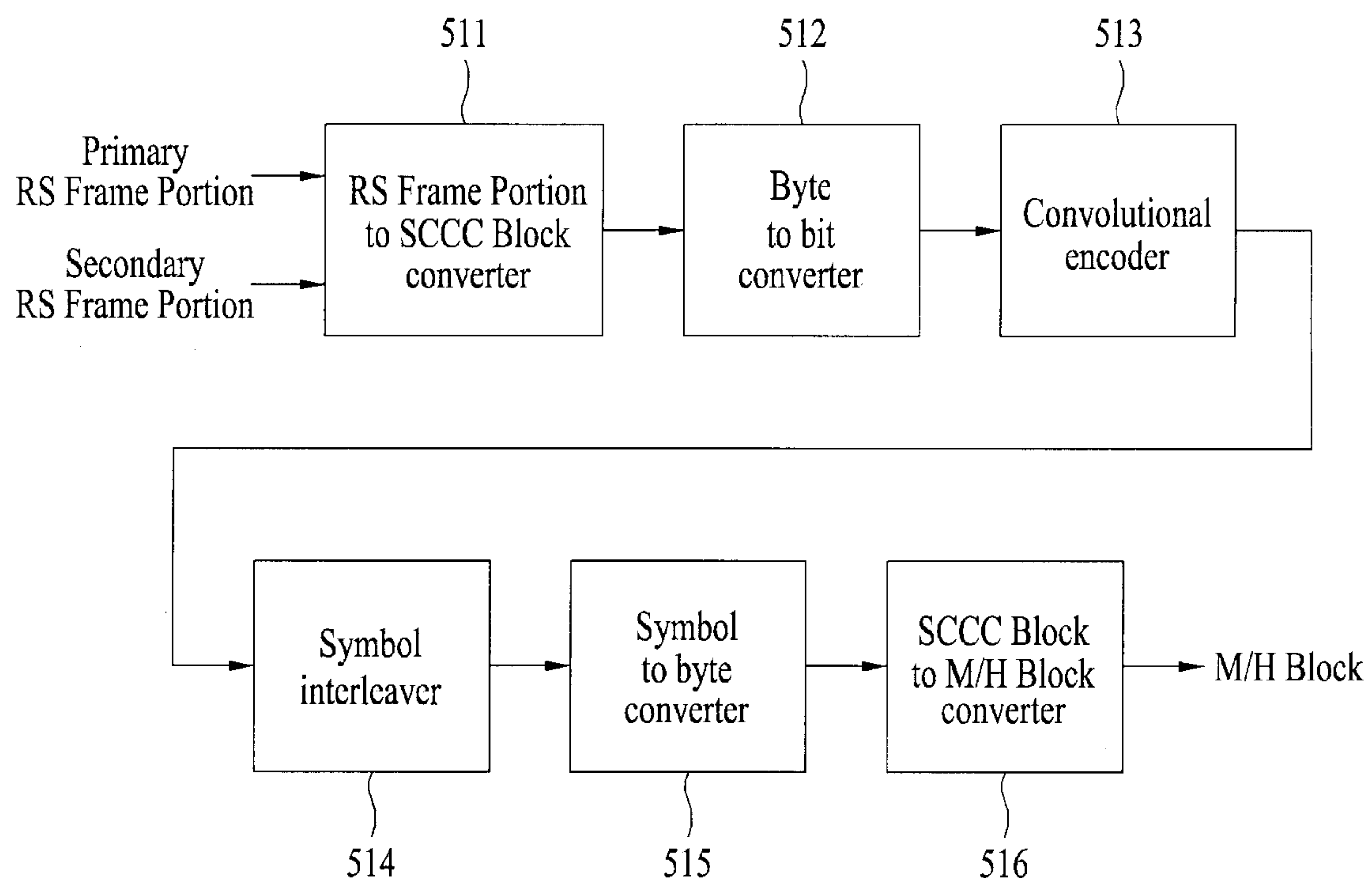


FIG. 23

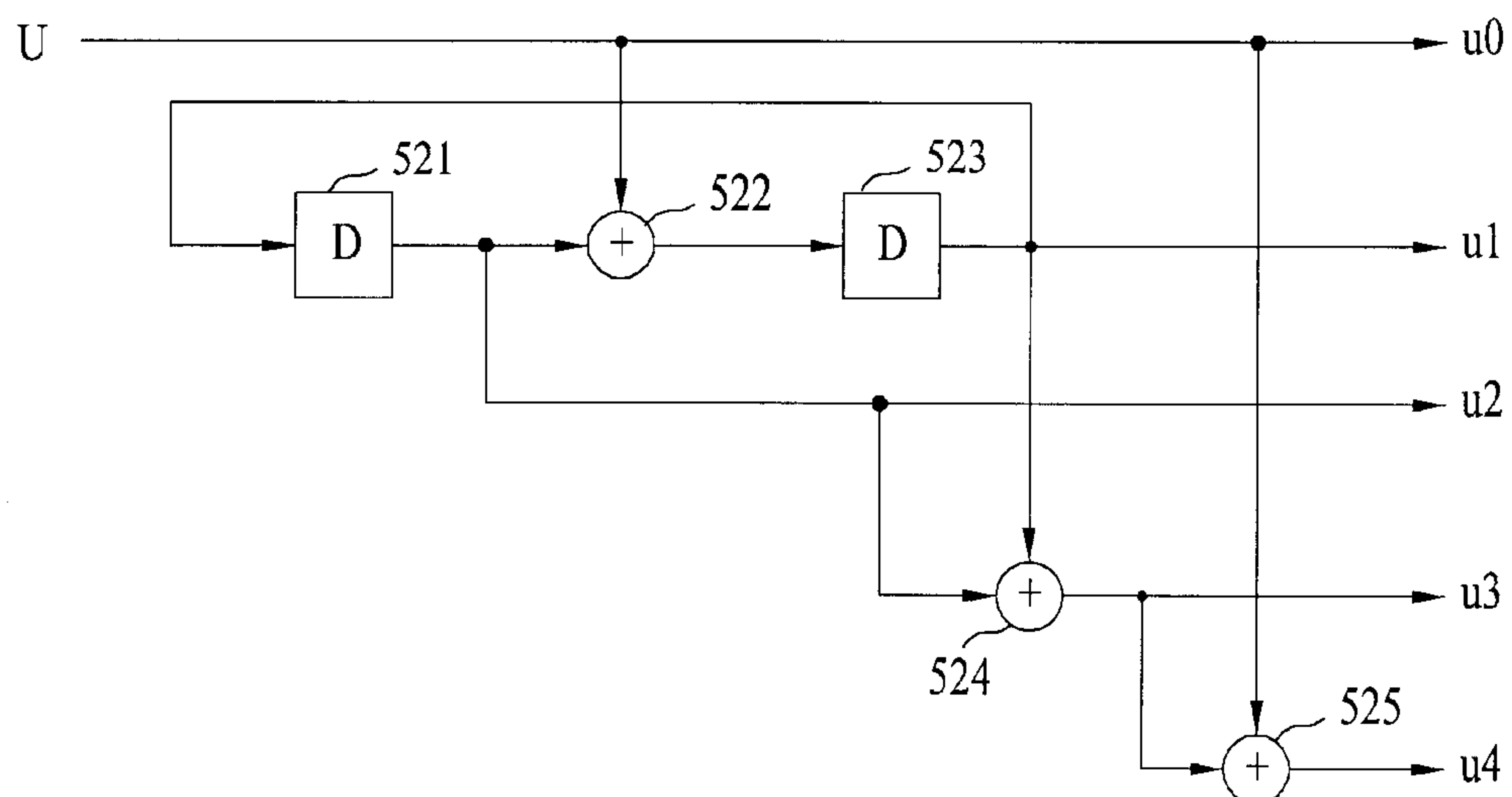


FIG. 24

| i     | 0 | 1  | 2   | 3   | 4   | 5    | 6    | 7    | 8    | 9    | 10   | 11   | 12   | 13   | 14   | 15   | ... | 2110 | 2111 |
|-------|---|----|-----|-----|-----|------|------|------|------|------|------|------|------|------|------|------|-----|------|------|
| P'(i) | 0 | 89 | 267 | 534 | 890 | 1335 | 1869 | 2492 | 3204 | 4005 | 799  | 1778 | 2846 | 4003 | 1153 | 2488 | ... | 2809 | 2272 |
| P(i)  | 0 | 89 | 267 | 534 | 890 | 1335 | 1869 | 799  | 1778 | 1153 | 1329 | 526  | 79   | 2037 | 253  | 874  | ... | 444  | 2048 |

FIG. 25

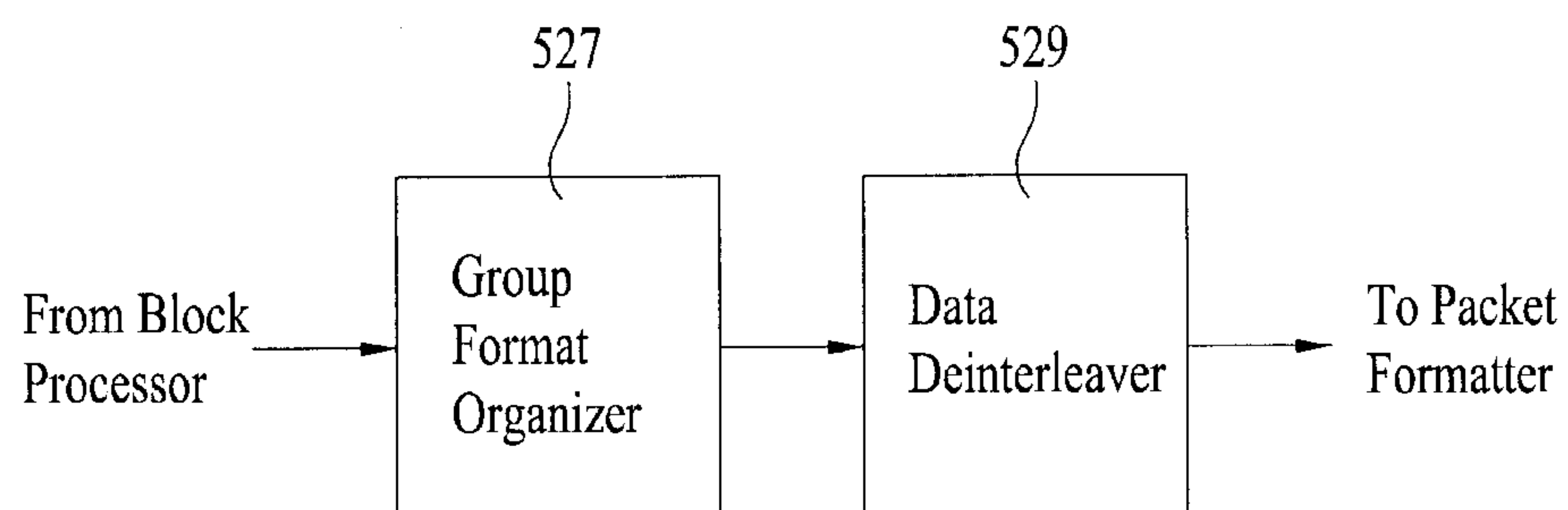
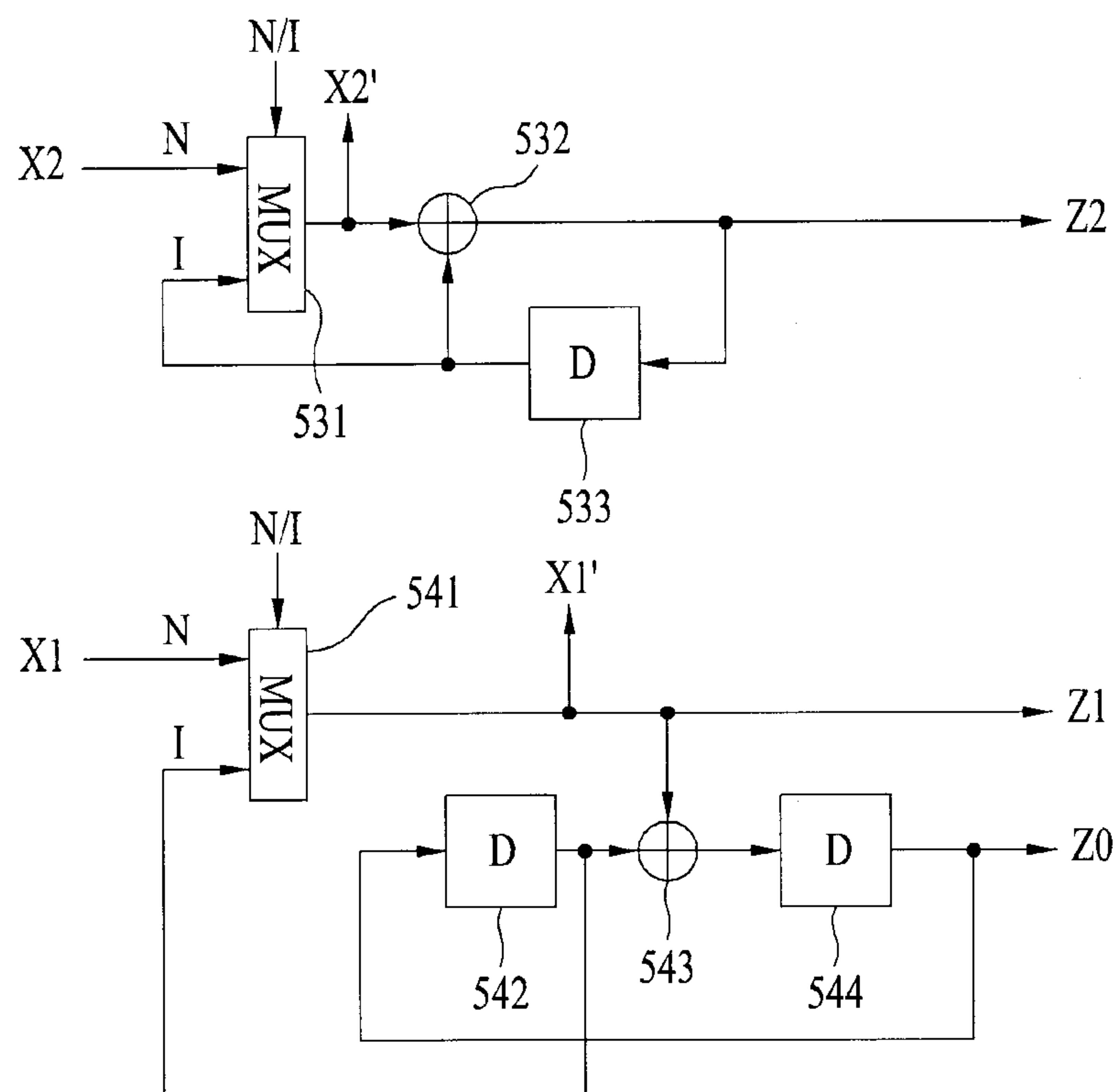


FIG. 26



$N$  : Normal mode,  $I$  : Initialization mode

FIG. 27

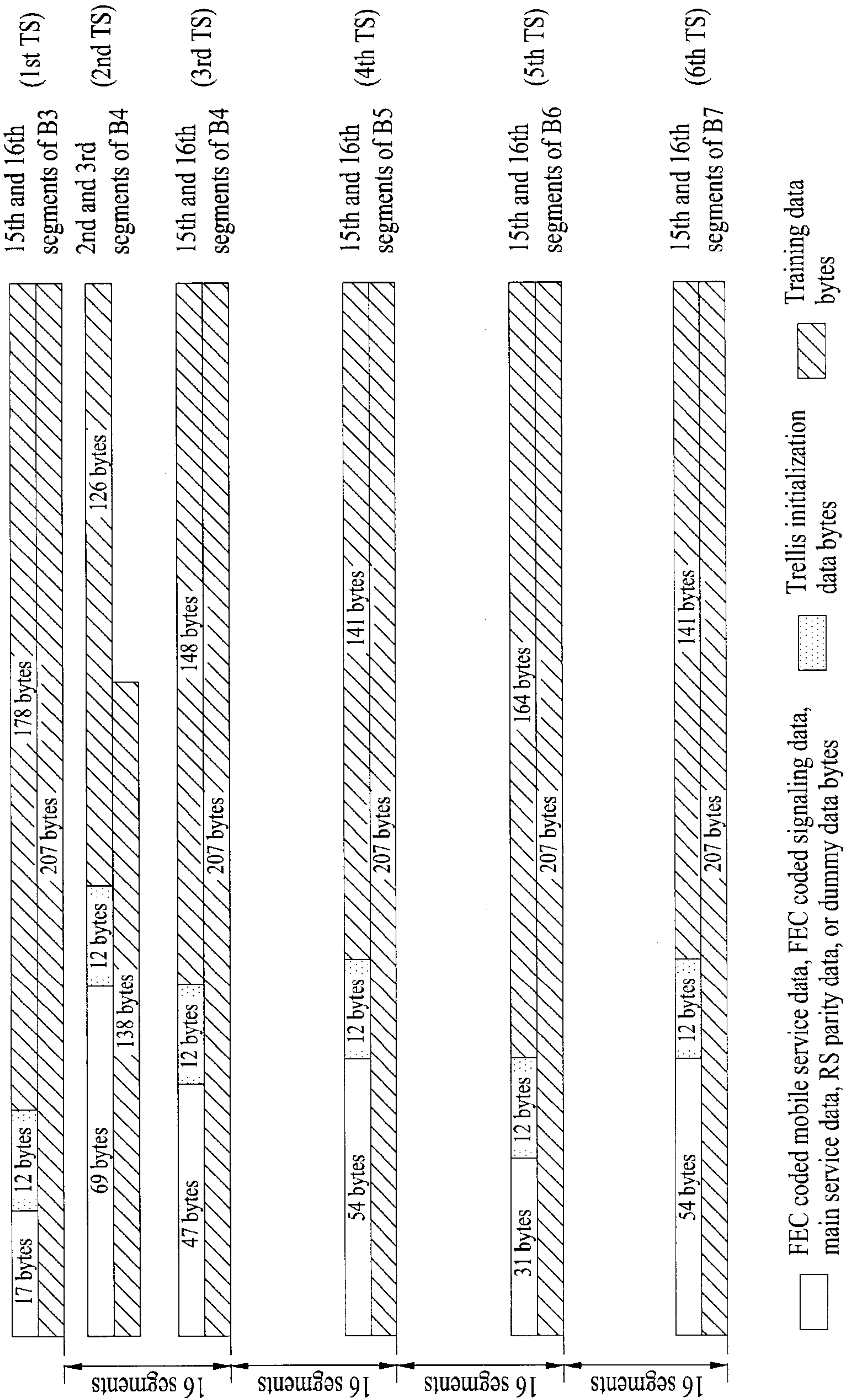


FIG. 28

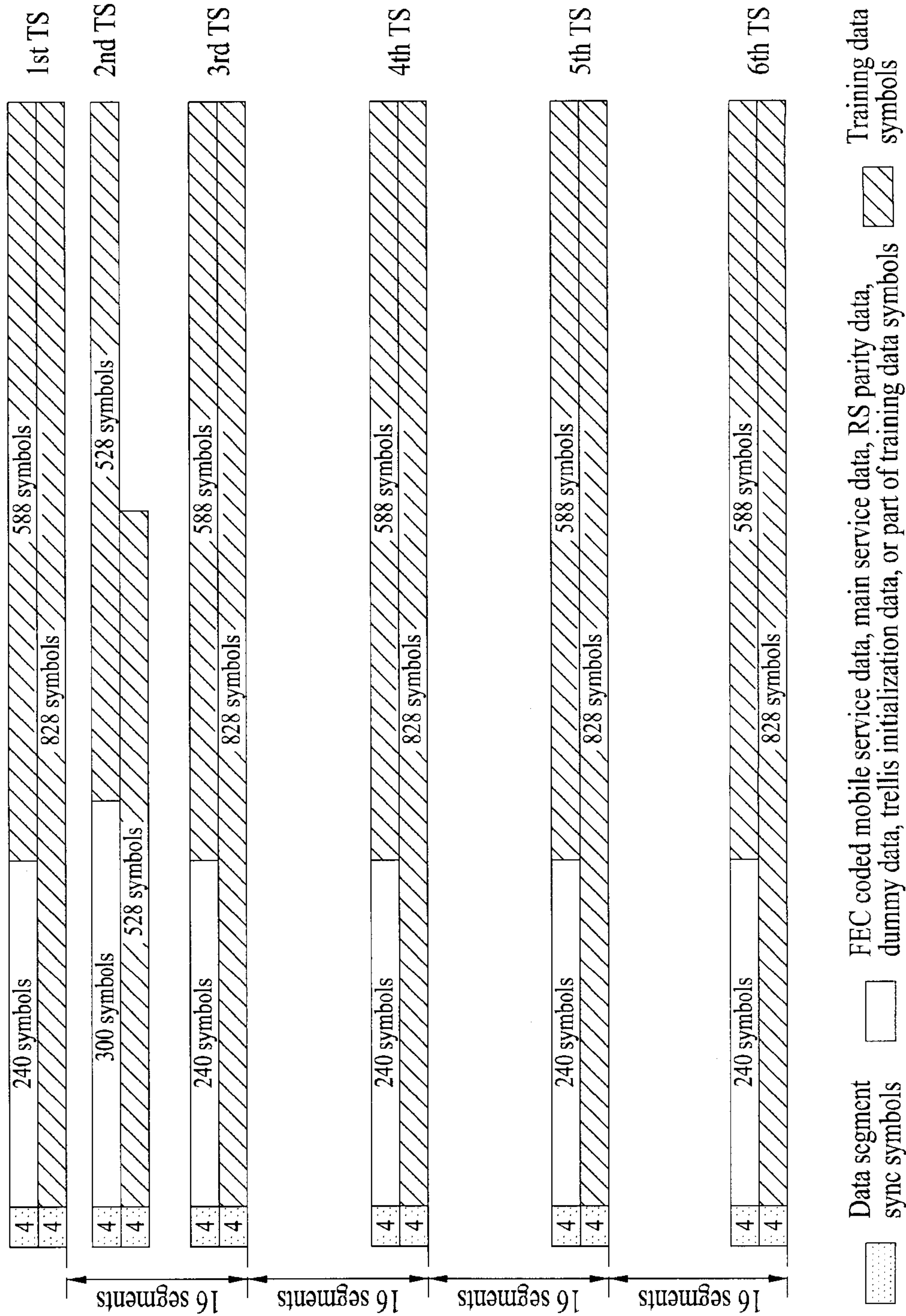


FIG. 29

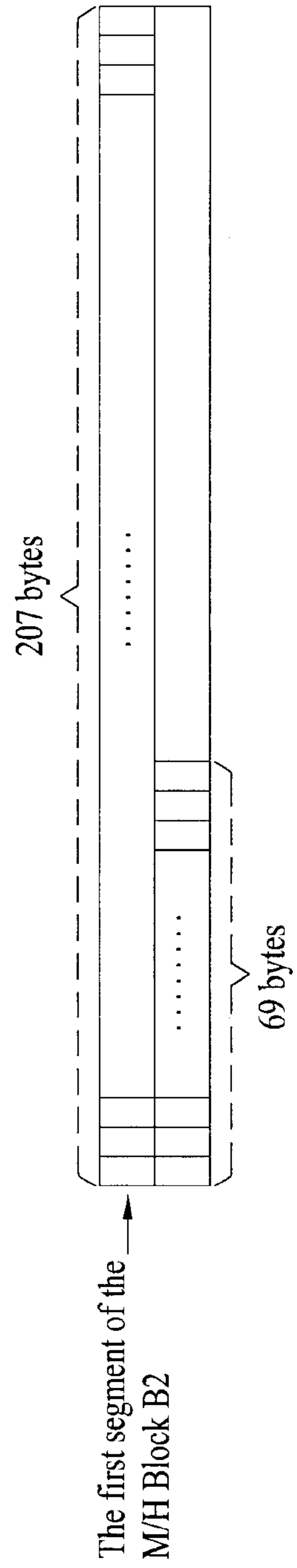


FIG. 30

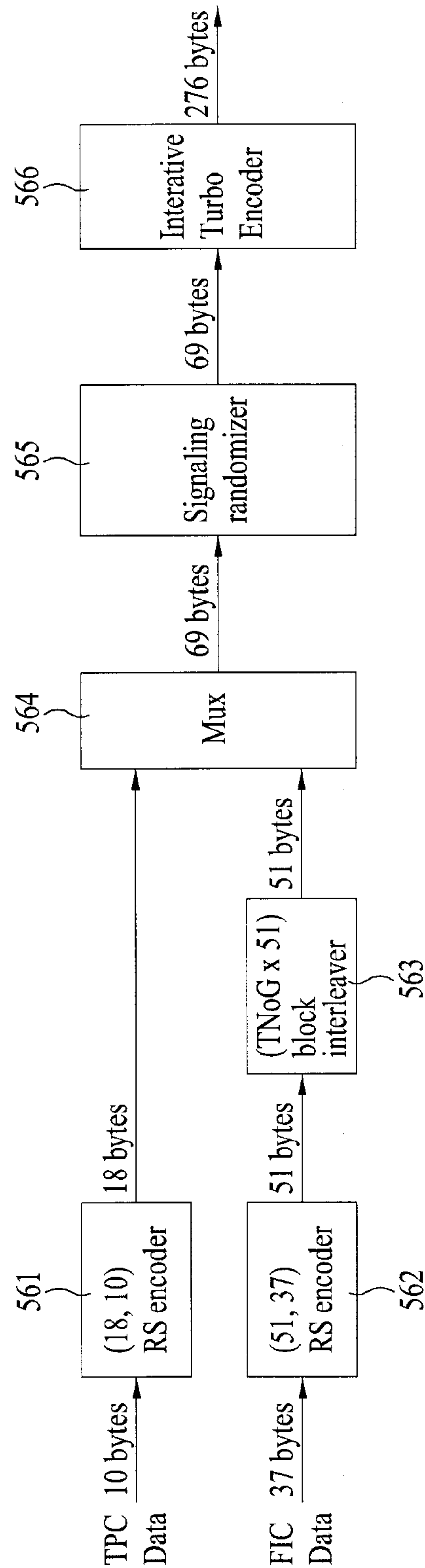


FIG. 31

| Syntax                    | No. of bits | Format |
|---------------------------|-------------|--------|
| TPC_data {                |             |        |
| Sub-Frame_number          | 3           | uimsbf |
| Slot_number               | 4           | uimsbf |
| Parade_id                 | 7           | uimsbf |
| starting_Group_number     | 4           | uimsbf |
| number_of_Groups          | 3           | uimsbf |
| Parade_repetition_cycle   | 3           | uimsbf |
| RS_Frame_mode             | 2           | bslbf  |
| RS_code_mode_primary      | 2           | bslbf  |
| RS_code_mode_secondary    | 2           | bslbf  |
| SCCC_Block_mode           | 2           | bslbf  |
| SCCC_outer_code_mode_A    | 2           | bslbf  |
| SCCC_outer_code_mode_B    | 2           | bslbf  |
| SCCC_outer_code_mode_C    | 2           | bslbf  |
| SCCC_outer_code_mode_D    | 2           | bslbf  |
| FIC_version               | 5           | uimsbf |
| Parade_continuity_counter | 4           | uimsbf |
| TNoG                      | 5           | uimsbf |
| reserved                  | 26          | bslbf  |
| }                         |             |        |

FIG. 32

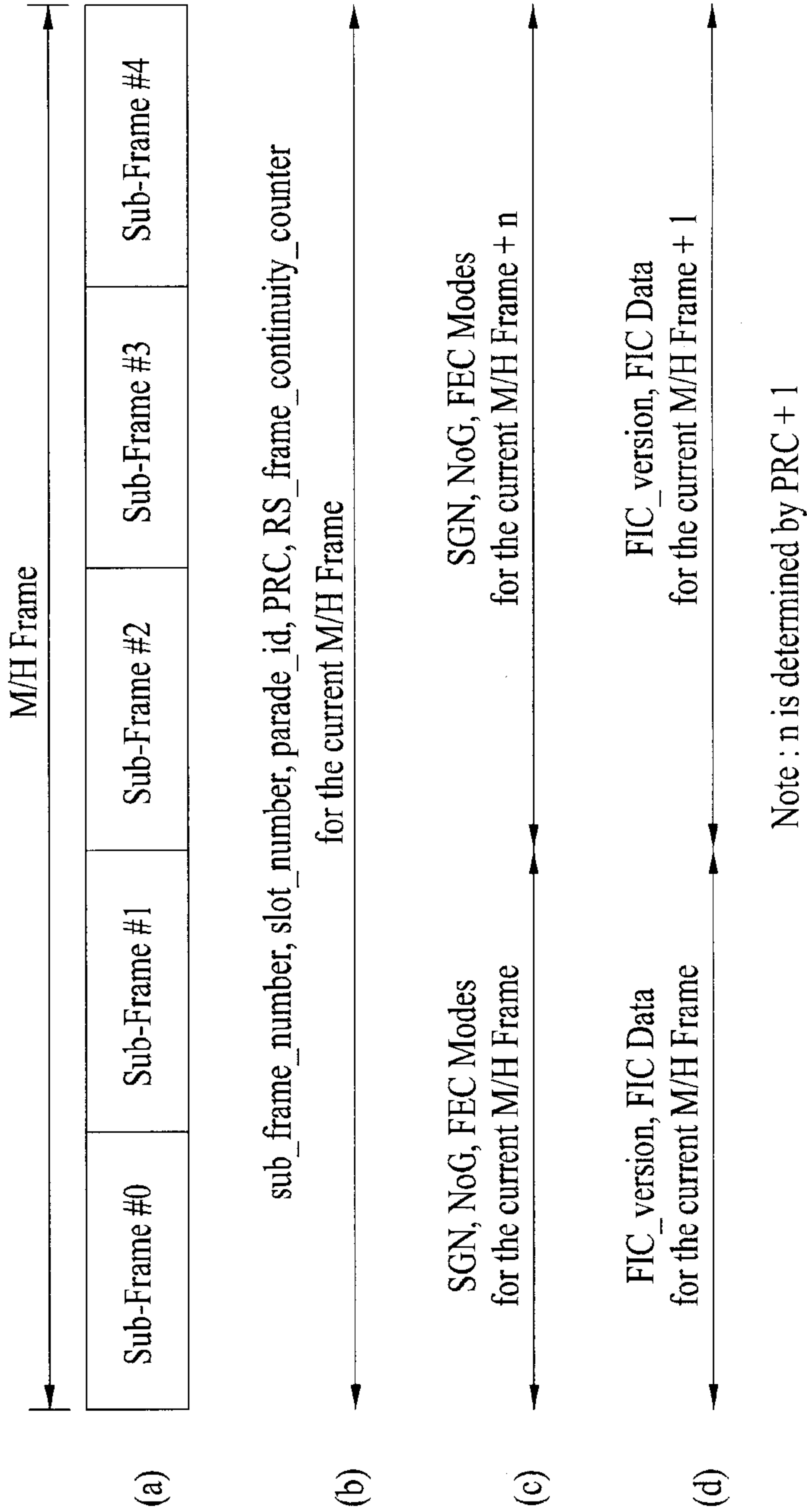


FIG. 33

| Syntax                                                          | No. of Bits |
|-----------------------------------------------------------------|-------------|
| FIC_chunk() {<br>FIC_chunk_header()<br>FIC_chunk_payload()<br>} | 5*8<br>var  |

FIG. 34

| Syntax                                                                                                                                                                                                                                                                                | No. of Bits                                | Format                                                                             |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|------------------------------------------------------------------------------------|
| FIC_chunk_header() {<br>FIC_major_protocol_version<br>FIC_minor_protocol_version<br>FIC_chunk_header_extension_length<br>ensemble_loop_header_extension_length<br>MH_service_loop_extension_length<br>reserved<br>current_next_indicator<br>transport_stream_id<br>num_ensembles<br>} | 2<br>3<br>3<br>3<br>3<br>1<br>1<br>16<br>8 | uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>'1'<br>bsblf<br>uimsbf<br>uimsbf |

FIG. 35

| Syntax                                                                                                                                                                                                                                                                                                                                                                                                                                                   | No. of Bits                                                                       | Format                                                                                                                                 |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| <pre> FIC_chunk_payload() {   for (i=0; i&lt;num_ensembles; i++) {     ensemble_id     reserved     ensemble_protocol_version     SLT_ensemble_indicator     GAT_ensemble_indicator     reserved     MH_service_signaling_channel_version     num_MH_services     for(j=0;j&lt;num_MH_services;j++){       MH_service_id       reserved       multi_ensemble_service       MH_service_status       SP_indicator     }   }   FIC_chunk_stuffing() }</pre> | <br>8<br>3<br>5<br>1<br>1<br>1<br>5<br>8<br><br>16<br>3<br>2<br>2<br>1<br><br>var | <br>uimsbf<br>'111'<br>uimsbf<br>bslbf<br>bslbf<br>'1'<br>uimsbf<br>uimsbf<br><br>uimsbf<br>'111'<br>uimsbf<br>uimsbf<br>bslbf<br><br> |

FIG. 36

| Syntax                                                                                                                                                                                    | No. of Bits                         | Format                                                             |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|--------------------------------------------------------------------|
| <pre> FIC_segment_header() {   FIC_segment_type   reserved   FIC_chunk_major_protocol_version   current_next_indicator   error_indicator   FIC_segment_num   FIC_last_segment_num }</pre> | <br>2<br>2<br>2<br>1<br>1<br>4<br>4 | <br>uimsbf<br>'11'<br>uimsbf<br>bslbf<br>bslbf<br>uimsbf<br>uimsbf |



FIG. 38

| Syntax                                  | No.of Bits | Format |
|-----------------------------------------|------------|--------|
| MH_component_descriptor() {             |            |        |
| descriptor_tag                          | 8          | 0 x CB |
| descriptor_length                       | 8          | uimsbf |
| component_type                          | 7          | uimsbf |
| component_encryption_flag               | 1          |        |
| if (component_encryption_flag == '1') { |            |        |
| num_STKM_streams                        | 8          | uimsbf |
| for (i=0; i<num_STKM_streams; i++) {    |            |        |
| STKM_stream_id                          | 8          | uimsbf |
| }                                       |            |        |
| transport_parameters_text_length        | 8          |        |
| transport_parameters_text()             | var        |        |
| MH_component_data(component_type)       | var        |        |
| }                                       |            |        |
| }                                       |            |        |

FIG. 39

| Syntax                           | Bits | Format |
|----------------------------------|------|--------|
| FIC-Segment_header () {          |      |        |
| FIC_segment_type                 | 2    | '00'   |
| Wake_up_indicator                | 1    | uimsbf |
| EAS_indicator                    | 1    | uimsbf |
| FIC_chunk_major_protocol_version | 2    | uimsbf |
| current_next_indicator           | 1    | uimsbf |
| error_indicator                  | 1    | uimsbf |
| FIC_segment_num                  | 4    | uimsbf |
| FIC_last_segment_num             | 4    | uimsbf |
| }                                |      |        |

FIG. 40

| Syntax                           | Bits | Format |
|----------------------------------|------|--------|
| FIC -Segment_header () {         |      |        |
| FIC_segment_type                 | 2    | '00'   |
| wake_up_indicator                | 2    | uimsbf |
| FIC_chunk_major_protocol_version | 2    | uimsbf |
| current_next_indicator           | 1    | uimsbf |
| error_indicator                  | 1    | uimsbf |
| FIC_segment_num                  | 4    | uimsbf |
| FIC_last_segment_num             | 4    | uimsbf |
| }                                |      |        |

FIG. 41

| wake_up_indicator | Description                  |
|-------------------|------------------------------|
| '00'              | No specified                 |
| '01'              | Wake-up information (Toggle) |
| '10'              |                              |
| '11'              | Reserved (General A/V)       |

FIG. 42

| Syntax                                                                                                                                                                                                                                                                                                                                | Bits                                                   | Format                                                                                         |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------|------------------------------------------------------------------------------------------------|
| FIC -chunk_payload () {<br>for (i=0; i<num_ensembles;i++) {<br>ensemble_id<br>reserved<br>ensemble_protocol_version<br>SLT_ensemble_indicator<br>GAT_ensemble_indicator<br>EAS_ensemble_indicator<br>MH_service_signaling_channel_version<br>num_MH_services<br>for(j= 0;j<num_MH_services;j++){<br>FIC service loop()<br>}<br>}<br>} | <br><br>8<br>3<br>5<br>1<br>1<br>1<br>5<br>8<br><br>24 | <br><br>uimsbf<br>'111'<br>uimsbf<br>bslbf<br>bslbf<br>bslbf<br>uimsbf<br>uimsbf<br><br>uimsbf |

FIG. 43

| Syntax                                                                                                                                                                    | No.of Bits                 | Format                                  |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|-----------------------------------------|
| MH_EAS_Message_descriptor() {<br>descriptor_tag<br>descriptor_length<br>length_of_CAP<br>for (i=0; i<length_of_CAP; i++)<br>{<br>CAP(Multiple String Structure)<br>}<br>} | <br>8<br>8<br>8<br><br>var | <br>0 x TBD<br>uimsbf<br>uimsbf<br><br> |

FIG. 44

| Syntax                                                                                                                                                                                                                                                                                                                                                                                                              | No.of Bits                                                                                | Format                                                                                                                                 |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| MH_EAS_Message_descriptor() {<br>descriptor_tag<br>descriptor_length<br>reserved<br>CAP_version_number<br>EAS_NRT_included_flag<br>if(EAS_NRT_included_flag){<br>reserved<br>Essential_Content_Linkage<br>num_associated_service<br>for (i=0; i<num_associated_service; i++)<br>{<br>associated_service_id<br>}<br>}<br>CAP_message_type<br>Length_of_CAP_message_id<br>CAP_message_id<br>Length_of_CAP<br>CAP<br>} | <br>8<br>16<br>3<br>4<br>1<br><br>4<br>8<br>4<br><br>16<br><br>1<br>7<br>var<br>16<br>var | <br>0 x TBD<br>uimsbf<br>'111'<br>uimsbf<br>bslbf<br><br>'1111'<br>uimsbf<br>uimsbf<br><br>uimsbf<br><br>bslbf<br>uimsbf<br><br>uimsbf |

FIG. 45

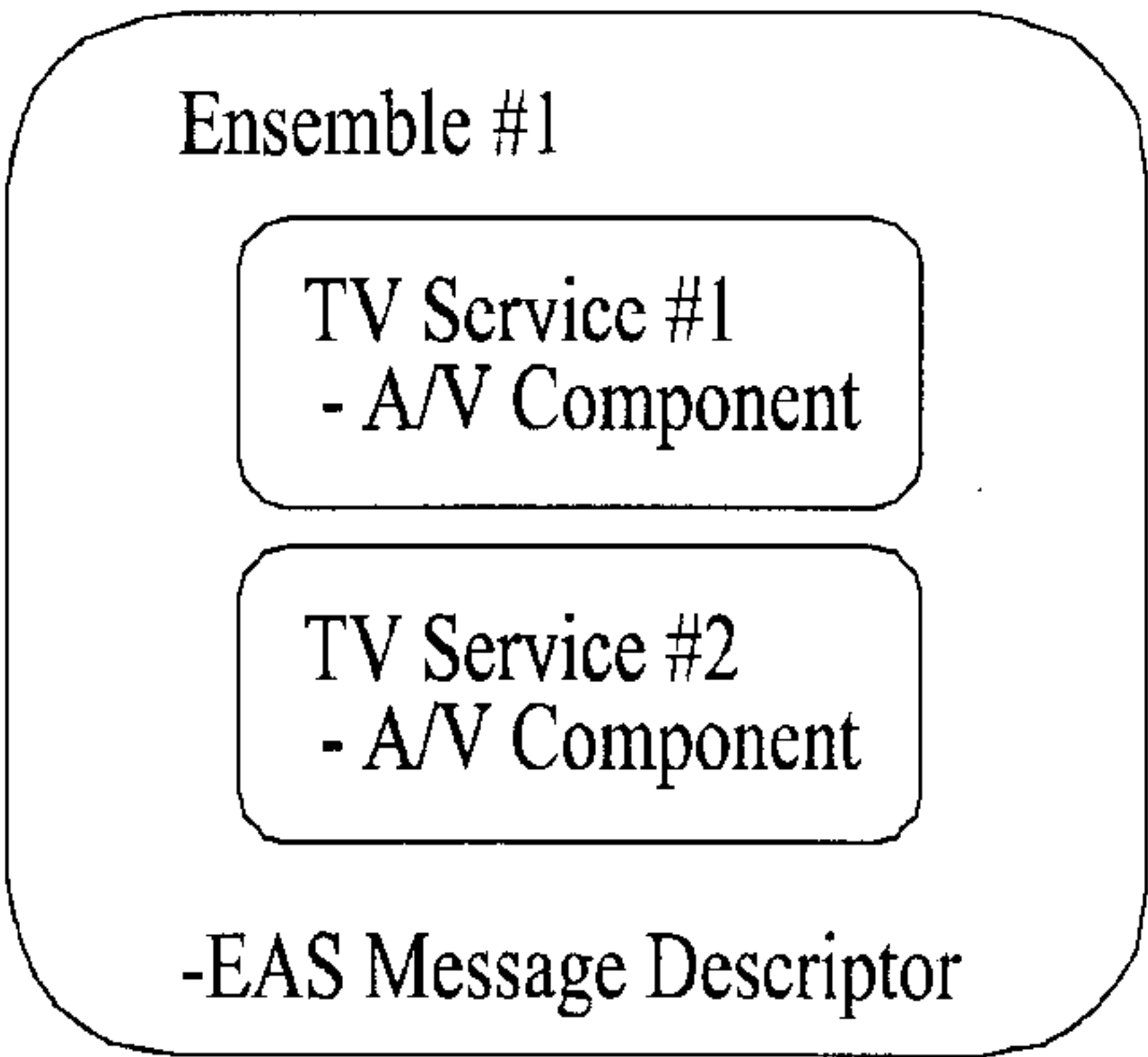


FIG. 46

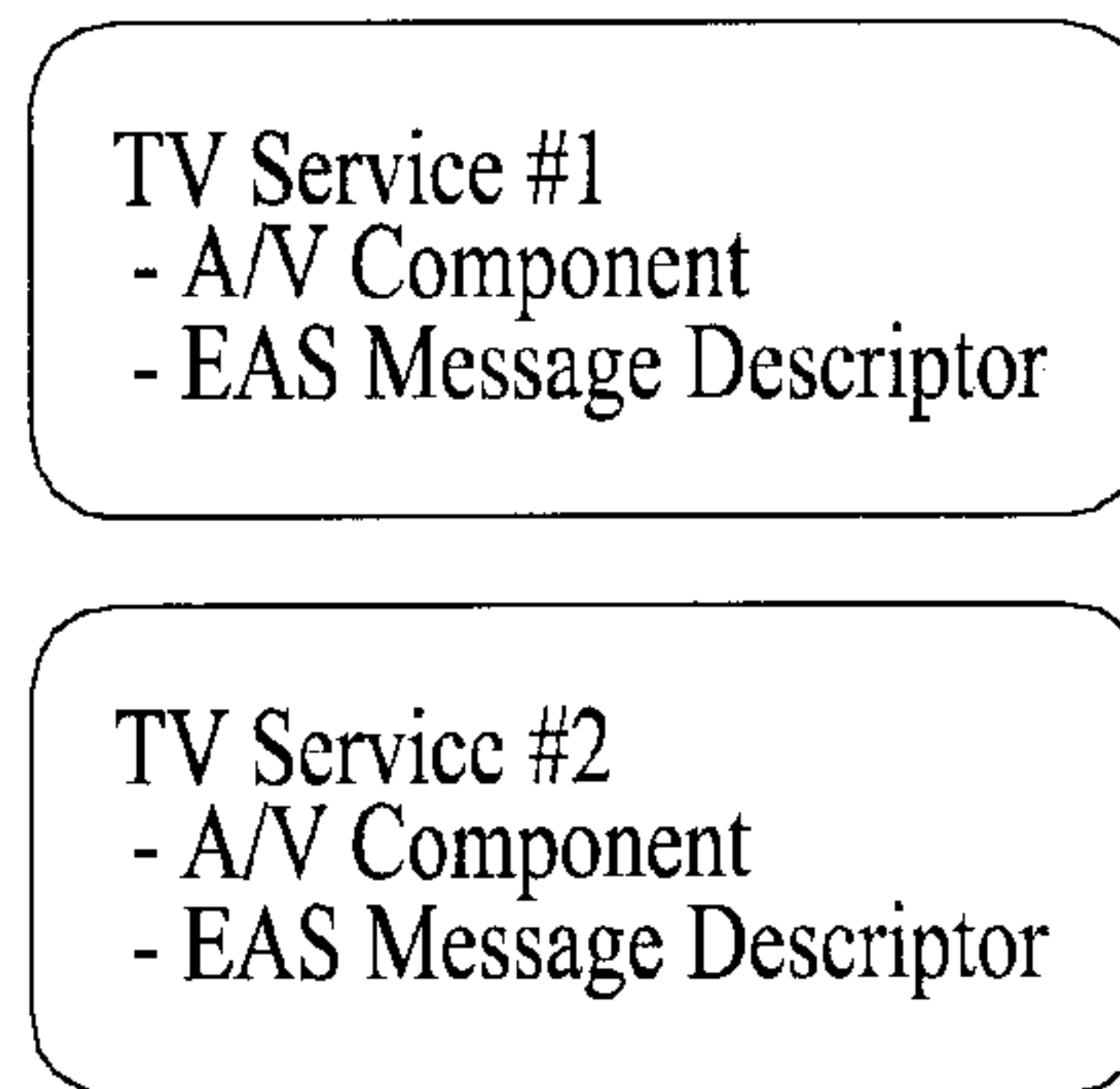


FIG. 47

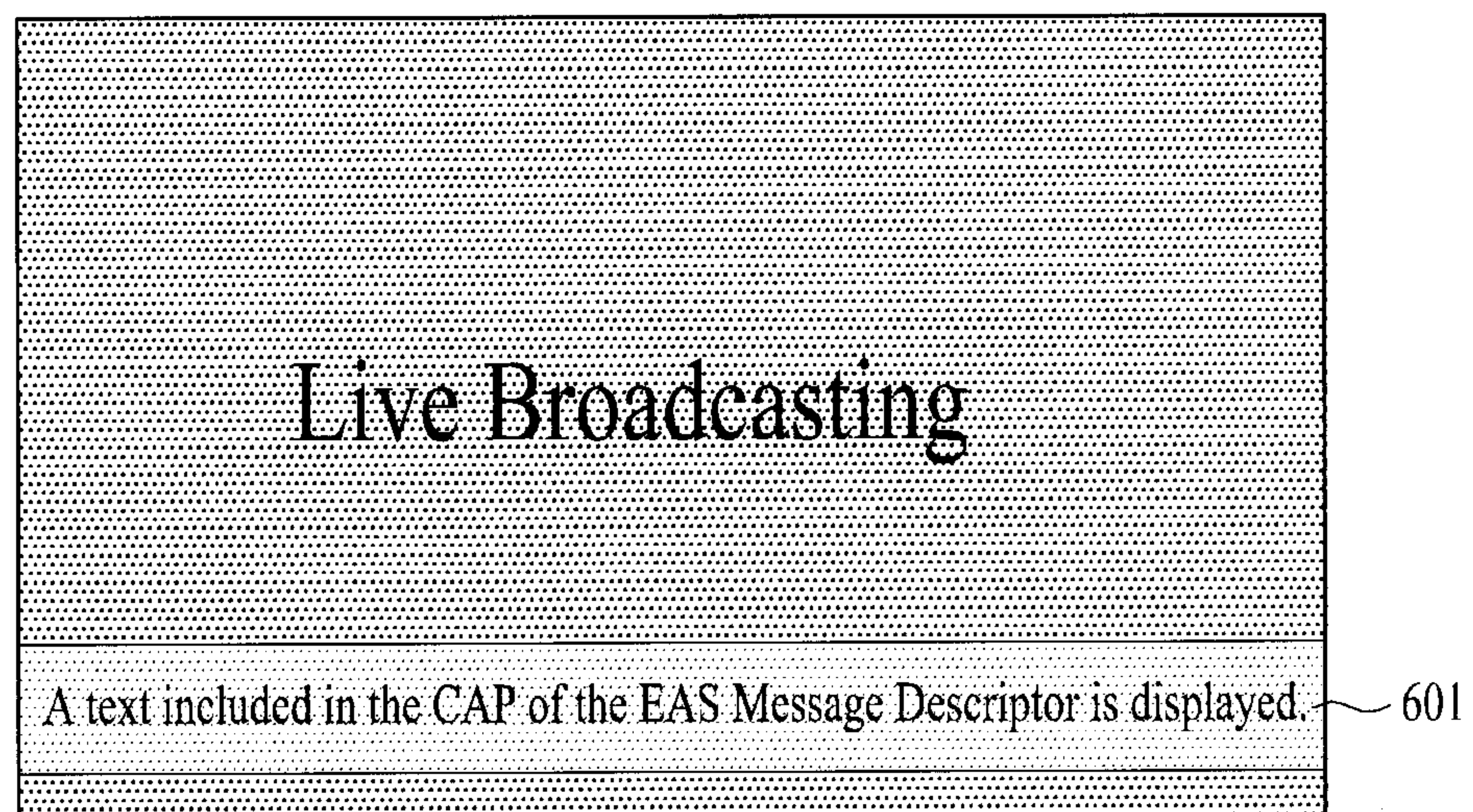


FIG. 48

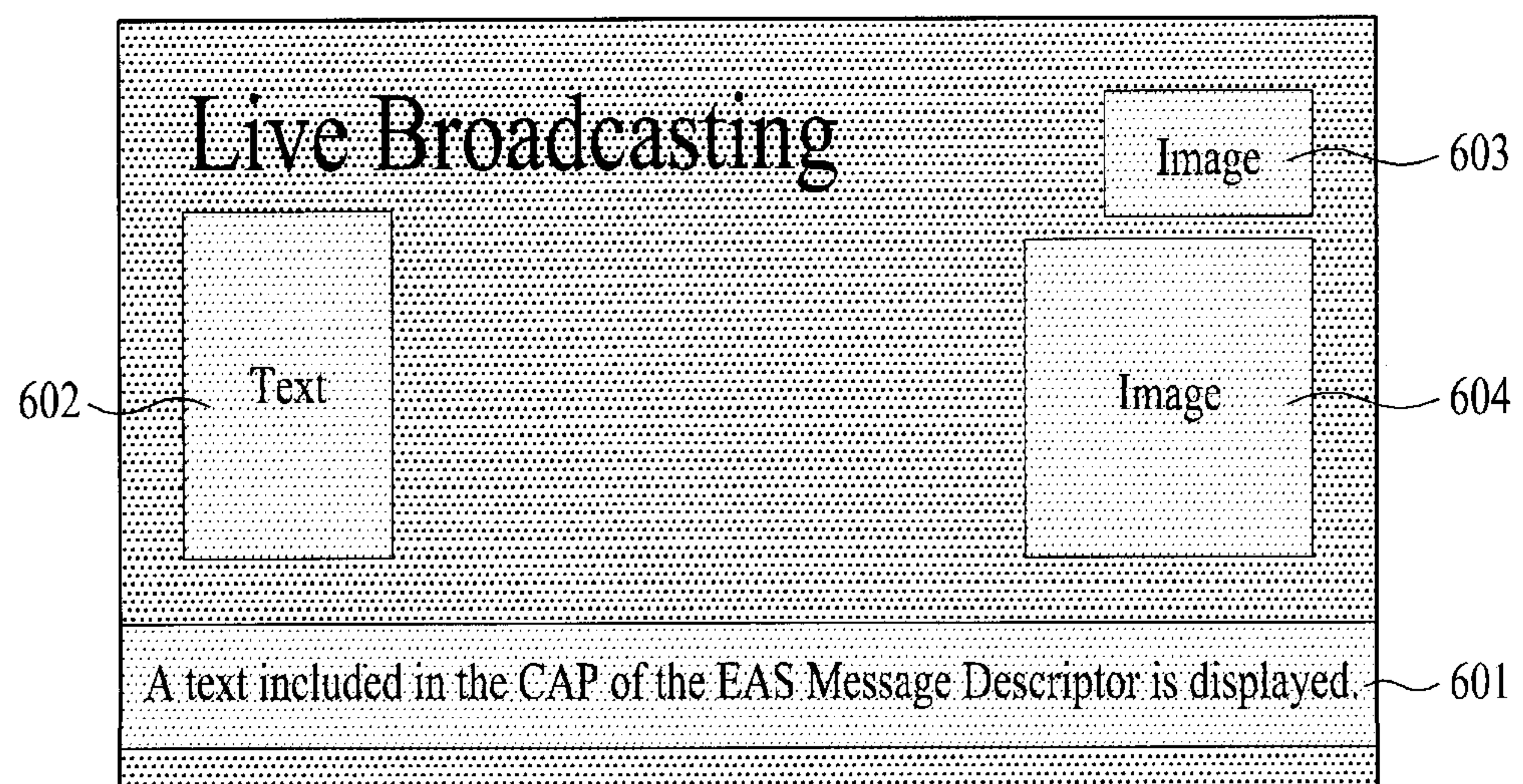


FIG. 49

| Syntax                                      | No. of Bits | Format |
|---------------------------------------------|-------------|--------|
| capabilities_descriptor() {                 |             |        |
| descriptor_tag                              | 8           | 0xC5   |
| descriptor_length                           | 8           | uimsbf |
| capability_code_count                       | 8           | uimsbf |
| for (i=0; i<capability_code_count; i++) {   |             |        |
| essential_indicator                         | 1           | bslbf  |
| capability_code                             | 7           | uimsbf |
| if (capability_code > 0x6F) {               |             |        |
| format_identifier                           | 32          | uimsbf |
| }                                           |             |        |
| }                                           |             |        |
| capability_string_count                     | 8           | uimsbf |
| for (i=0; i<capability_string_count; i++) { |             |        |
| essential_indicator                         | 1           |        |
| capability_category_code                    | 7           | uimsbf |
| capability_string_length                    | 8           | uimsbf |
| capability_string()                         | var         |        |
| }                                           |             |        |
| for (j=0; j<N; j++) {                       |             |        |
| reserved                                    | 8           | bslbf  |
| }                                           |             |        |
| }                                           |             |        |

Service : NRT Service #1

- NRT Service Descriptor
- NRT Capabilities Descriptor : x-application/eas
- Flute Component
  - text
  - image
  - image

FIG. 50

| Syntax                                 | No. of Bits | Format  |
|----------------------------------------|-------------|---------|
| NRT_service_descriptor() {             |             |         |
| descriptor_tag                         | 8           | 0xC4    |
| descriptor_length                      | 8           | uimbsf  |
| reserved                               | 2           | '11'    |
| consumption_model                      | 6           | uimbsf  |
| auto-update                            | 1           | bslbf   |
| storage_reservation_present            | 1           | bslbf   |
| default_content_size_present           | 1           | bslbf   |
| reserved                               | 5           | '11111' |
| if (storage_reservation_present==1) {  |             |         |
| storage_reservation                    | 24          | uimbsf  |
| if (default_content_size_present==1) { |             |         |
| default_content_size                   | 40          | uimbsf  |
| }                                      |             |         |
| for (j=0; j< N; j++) {                 |             |         |
| reserved                               | 8           | bslbf   |
| }                                      |             |         |
| }                                      |             |         |

FIG. 51

| consumption_model | Meaning                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0x00              | Forbidden.                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| 0x01              | Browse & Download - The NRT service describes content that can be selected for later download.                                                                                                                                                                                                                                                                                                                                                                          |
| 0x02              | Portal - The NRT service provides an experience similar to a web browser access. Files needed to support text/graphics rendering are available in the associated FLUTE session.                                                                                                                                                                                                                                                                                         |
| 0x03              | Push - The NRT service offers request-based content. Receivers are expected to offer the user a choice whether or not to automatically update content associated with the service. For such services, if the user selects the auto-update option, the receiver caches and service-related content and automatically updates files as new versions are made available. When the user returns to a requested Push service, content that had been pre-loaded is displayed. |
| 0x04              | EAS                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 0x05 - 0x3F       | Reserved for use by ATSC or other SDOs who register the use with ATSC.                                                                                                                                                                                                                                                                                                                                                                                                  |

FIG. 52

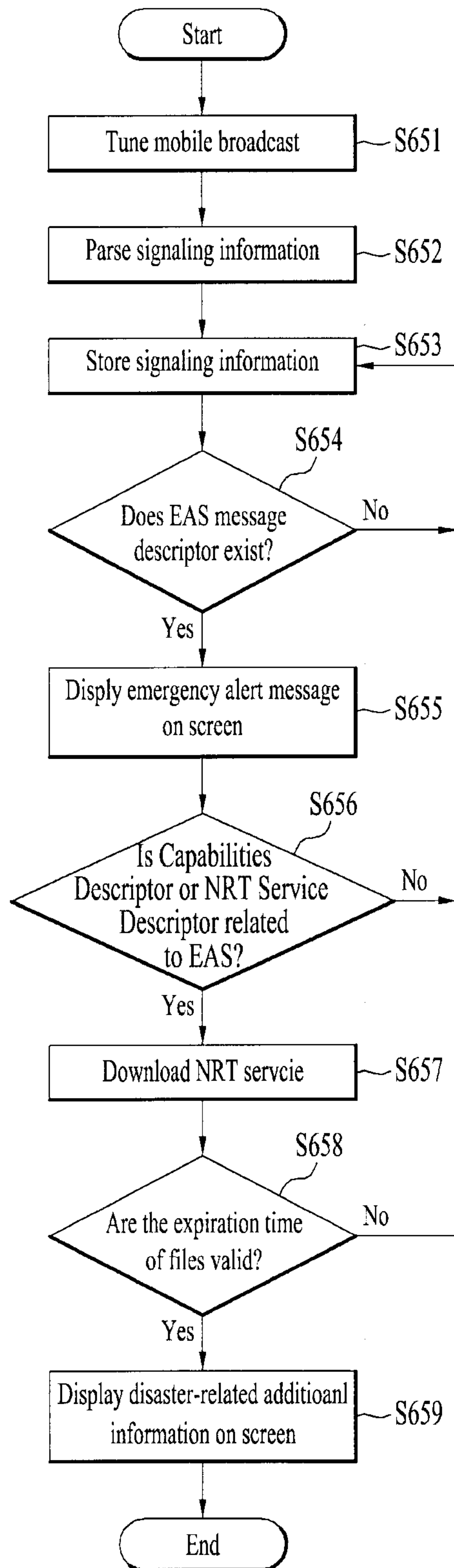


FIG. 53

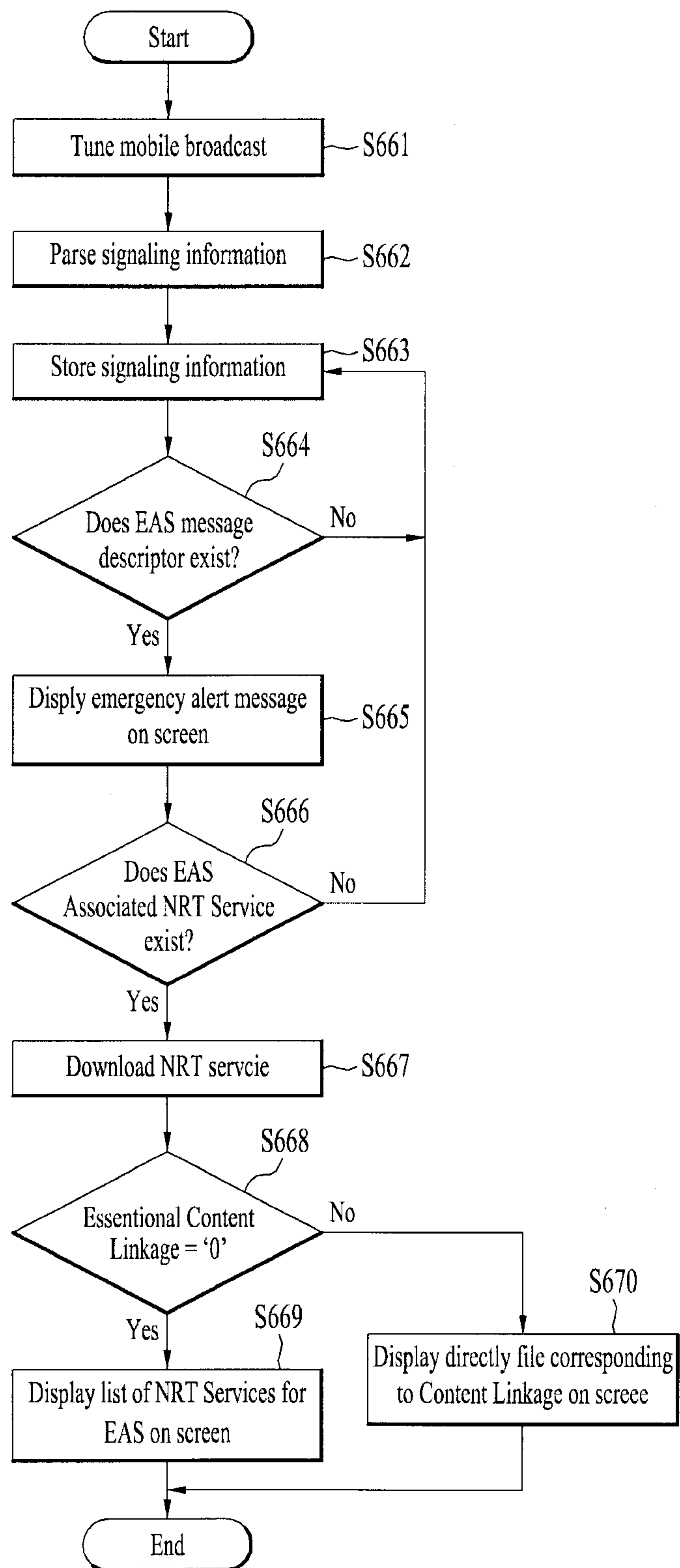


FIG. 54

|           |            |                                            |
|-----------|------------|--------------------------------------------|
| IP Header | UDP Header | Disaster broadcast Payload (CAP or Syntax) |
|-----------|------------|--------------------------------------------|

FIG. 55

| Syntax                                   | No.of Bits | Format                   |
|------------------------------------------|------------|--------------------------|
| emergency_alerting_system_descriptor() { |            |                          |
| descriptor_tag                           | 8          | uimsbf value '0xXX'      |
| descriptor_length                        | 8          | uimsbf                   |
| descriptor_number                        | 4          | uimsbf                   |
| last_descriptor_number                   | 4          | uimsbf                   |
| EAS_event_ID                             | 16         | uimsbf                   |
| EAS_originator_code                      | 24         | uimsbf three ASCII char  |
| EAS_event_code_length                    | 8          | uimsbf (N)               |
| EAS_event_code                           | 8*N        | uimsbf N ASCII char      |
| alerting_message_time_remaining          | 8          | uimsbf seconds (0~120)   |
| event_start_time                         | 32         | uimsbf                   |
| event_duration                           | 16         | uimsbf seconds (15~6000) |
| alerting_priority                        | 4          | uimsbf                   |
| EAS_major_channel_number                 | 8          | uimsbf                   |
| EAS_minor_channel_number                 | 8          | uimsbf                   |
| alerting_text_length                     | 16         | uimsbf                   |
| alerting_text()                          | var        | Per ATSC A/65-A, Sec6.8  |
| location_code_count                      | 8          | uimsbf (1~31)            |
| for (i=0; i<location_code_count; i++) {  |            |                          |
| state_code                               | 8          | uimsbf (0~99)            |
| county_subdivision                       | 4          | uimsbf (0~9)             |
| reserved                                 | 2          | '11'                     |
| county_code                              | 10         | uimsbf (0~999)           |
| }                                        |            |                          |
| }                                        |            |                          |

FIG. 56

| MH_service_category | Meaning                         |
|---------------------|---------------------------------|
| 0x00                | Not specified                   |
| 0x01                | Basic TV                        |
| 0x02                | Basic Radio                     |
| 0x03                | RI Service                      |
| 0x04                | EAS (Emergency Alerting System) |
| 0x05    0x07        | Not specified                   |
| 0x08                | Service Guide                   |
| 0x09    0x0A        | Not specified                   |
| 0x0B    0xFF        | Reserved for future use         |

FIG. 57

| Syntax                 | No. of Bits | Format |
|------------------------|-------------|--------|
| MH_component_data () { |             |        |
| EAS_event_ID           | 16          | uimsbf |
| payload_type           | 2           | uimsbf |
| status                 | 6           | uimsbf |
| urgency                | 8           | uimsbf |
| severity               | 8           | uimsbf |
| certainty              | 8           | uimsbf |
| event_start_time       | 32          | uimsbf |
| event_duration         | 16          | uimsbf |
| }                      |             |        |

FIG. 58

| Syntax                                   | No. of Bits | Format  |
|------------------------------------------|-------------|---------|
| Emergency_alerting_table_section() {     |             |         |
| table_id                                 | 8           | 0xEA    |
| section_syntax_indicator                 | 1           | '0'     |
| private_indicator                        | 1           | '1'     |
| reserved                                 | 2           | '11'    |
| section_length                           | 12          | uimsbf  |
| table_id_extension {                     |             |         |
| EAT_protocol_version                     | 8           | uimsbf  |
| ensemble_id                              | 8           | uimsbf  |
| }                                        |             |         |
| reserved                                 | 2           | '11'    |
| version_number                           | 5           | uimsbf  |
| current_next_indicator                   | 1           | bslbf   |
| section_number                           | 8           | uimsbf  |
| last_section_number                      | 8           | uimsbf  |
| num_MH_services                          | 8           | uimsbf  |
| for (i=0; i< num_MH_services; i++ )      |             |         |
| {                                        |             |         |
| MH_service_id                            | 16          | uimsbf  |
| MH_service_status                        | 2           | uimsbf  |
| SP_indicator                             | 1           | uimsbf  |
| reserved                                 | 5           | '11111' |
| urgency                                  | 8           | uimsbf  |
| severity                                 | 8           | uimsbf  |
| certainty                                | 8           | uimsbf  |
| event_start_time                         | 32          | uimsbf  |
| event_duration                           | 16          | uimsbf  |
| IP_version_flag                          | 1           | bslbf   |
| service_source_IP_address_flag           | 1           | bslbf   |
| service_destination_IP_address_flag      | 1           | bslbf   |
| IP_payload_type                          | 2           | uimsbf  |
| reserved                                 | 3           | '111'   |
| if(service_source_IP_address_flag)       |             |         |
| service_source_IP_address                | 32 or 128   | uimsbf  |
| if(service_destination_IP_address_flag)  |             |         |
| service_destination_IP_address           | 32 or 128   | uimsbf  |
| if(severity == 1) {                      |             |         |
| force_channel_num                        | 8           | uimsbf  |
| force_service_id                         | 16          | uimsbf  |
| }                                        |             |         |
| num_MH_EAS_descriptor                    | 8           | uimsbf  |
| for(j=0; j<num_MH_EAS_descriptor; j++) { |             |         |
| MH_EAS_descriptor()                      | var         |         |
| }                                        |             |         |
| }                                        |             |         |
| }                                        |             |         |

FIG. 59

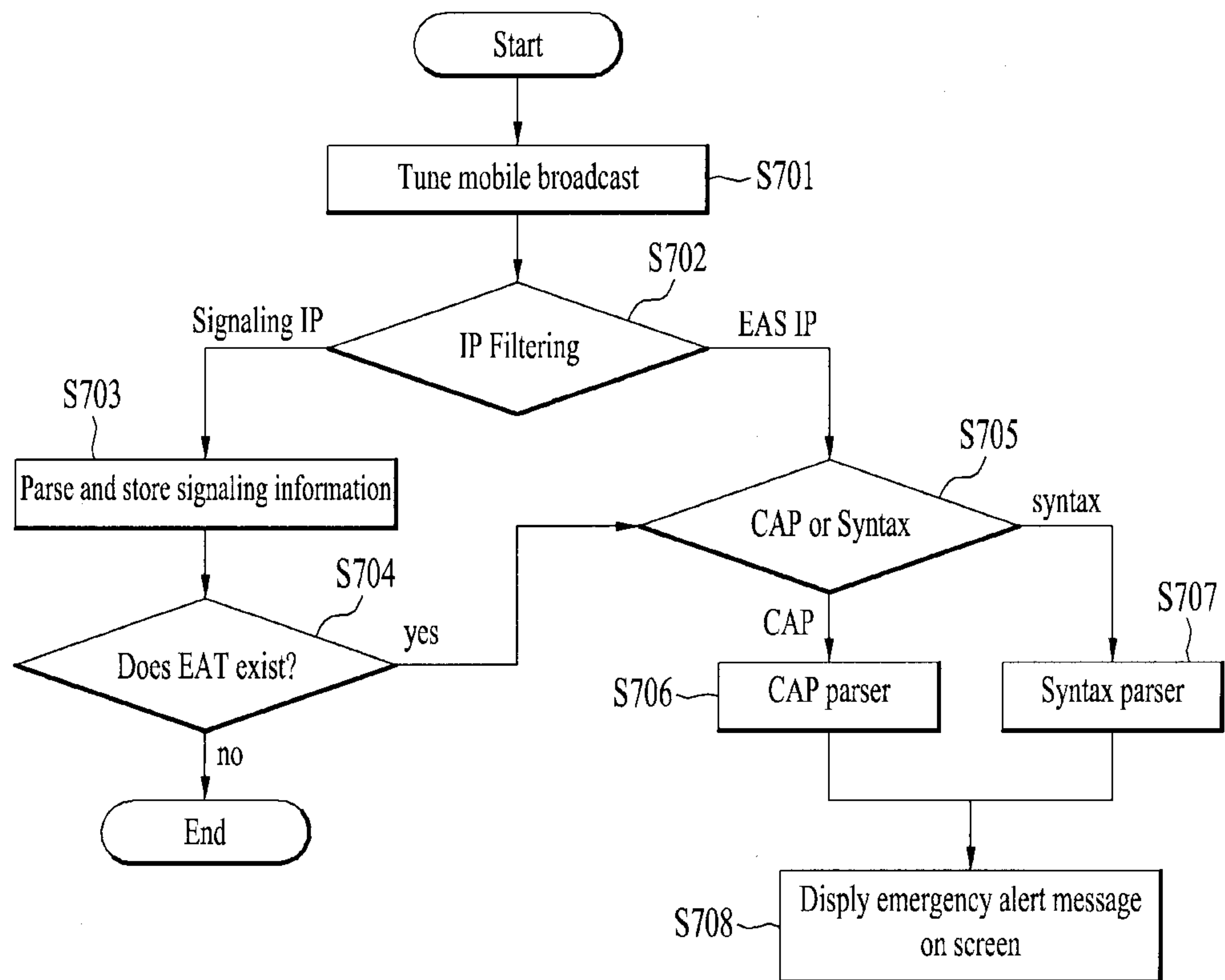
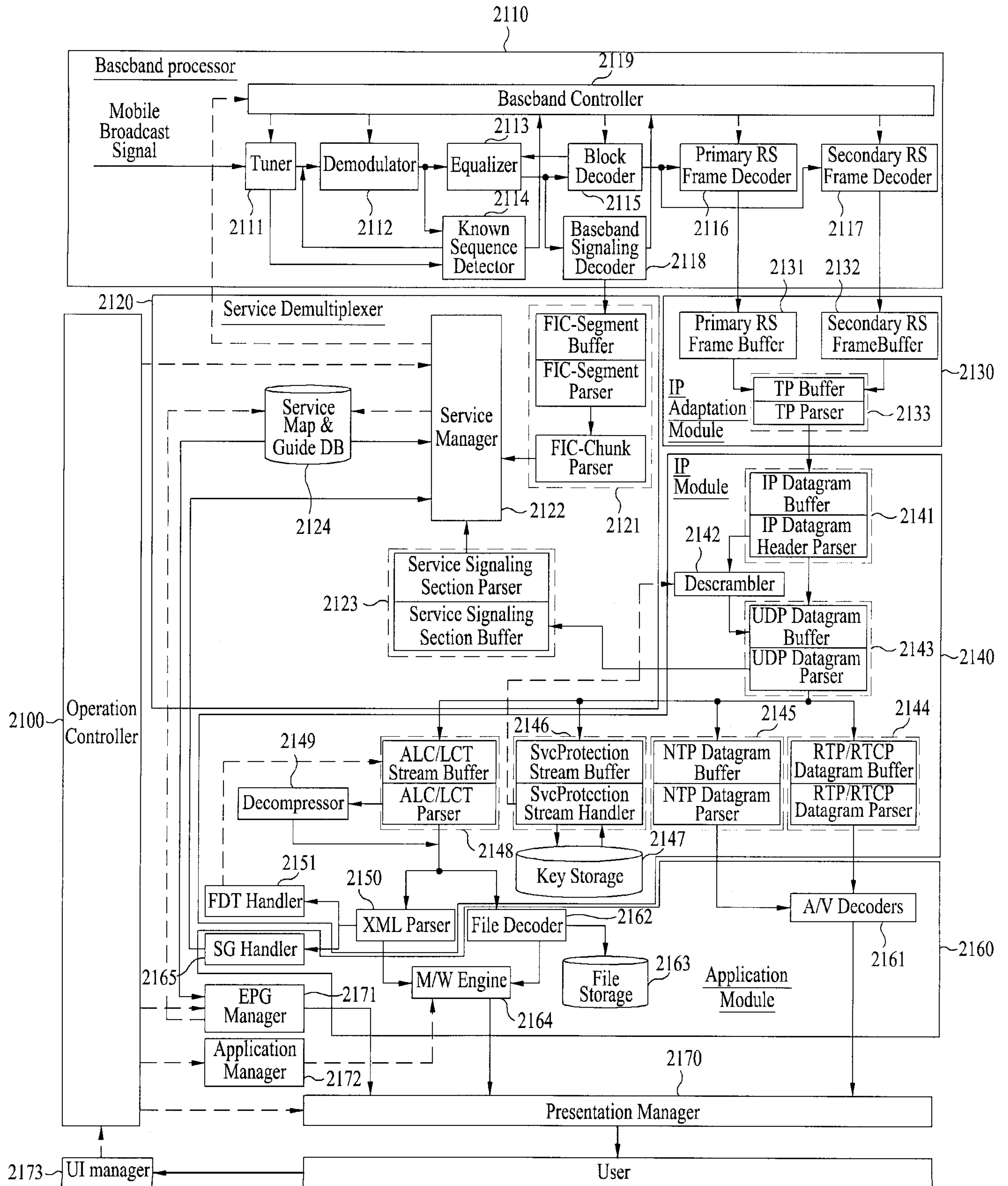


FIG. 60



| Syntax                                                                                                                                                                                             | Bits                            | Format                                                           |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------|------------------------------------------------------------------|
| FIC -Segment_header () {<br>FIC_segment_type<br>wake_up_indicator<br>FIC_chunk_major_protocol_version<br>current_next_indicator<br>error_indicator<br>FIC_segment_num<br>FIC_last_segment_num<br>} | 2<br>2<br>2<br>1<br>1<br>4<br>4 | '00'<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf<br>uimsbf |