

FIG.1

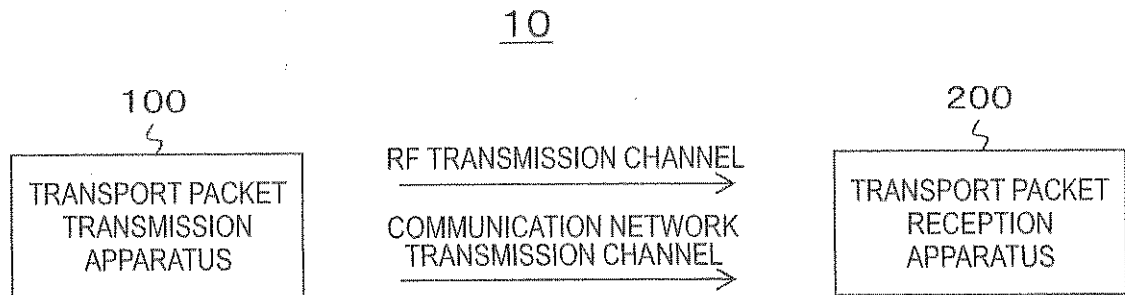
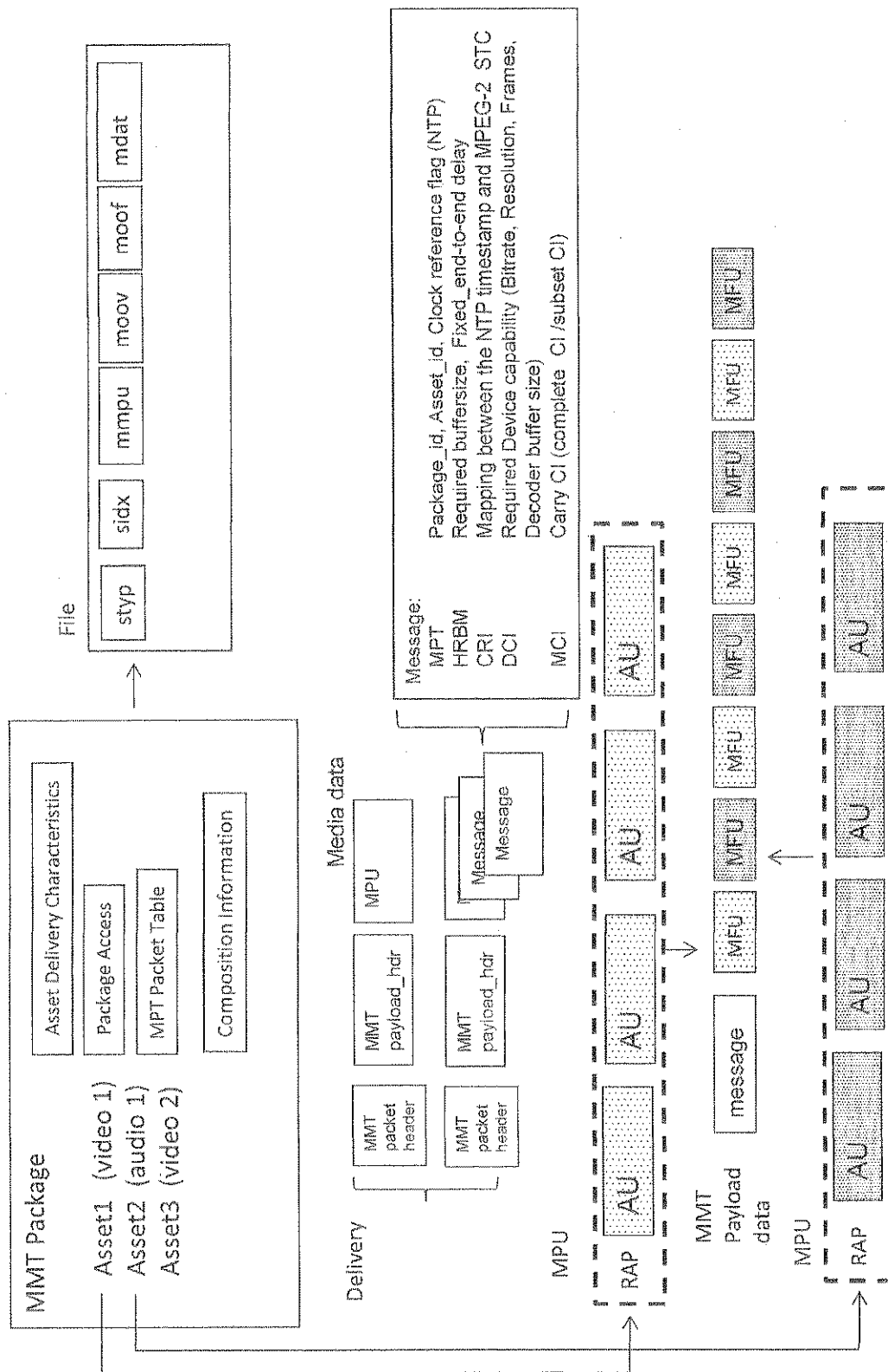
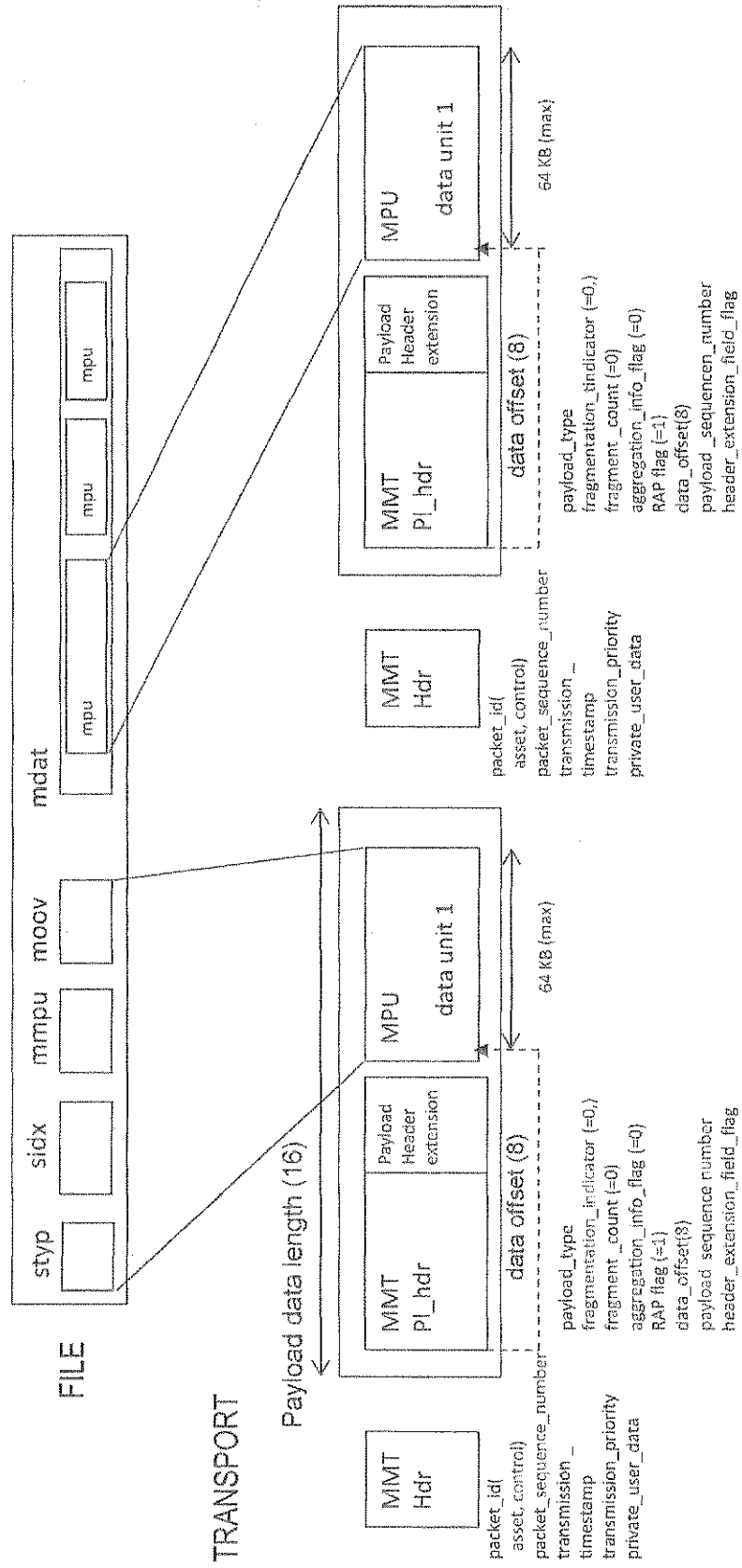


FIG.2





MPU packetize case1 "Meta, or MPU data in MMT payload"



५८५

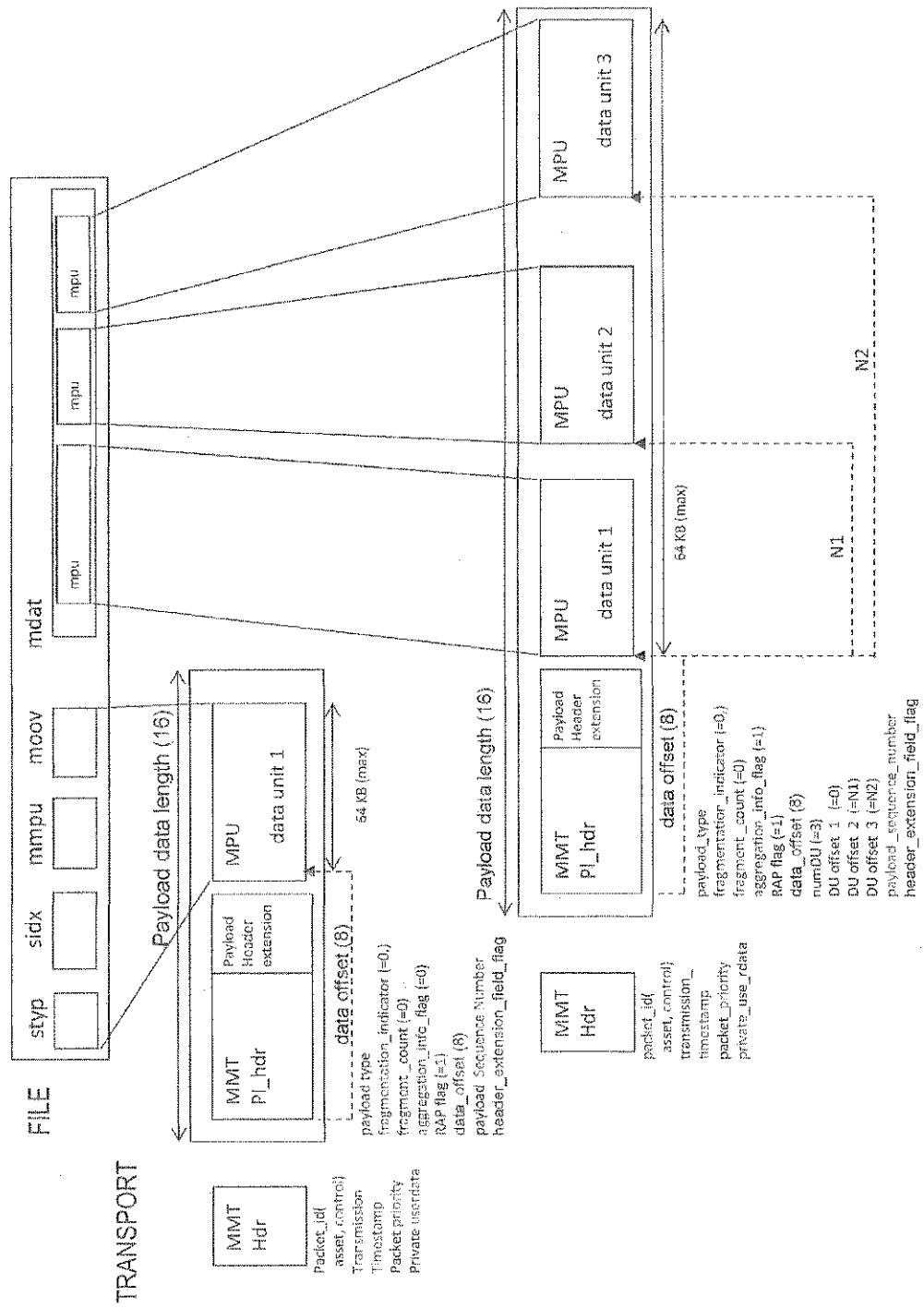
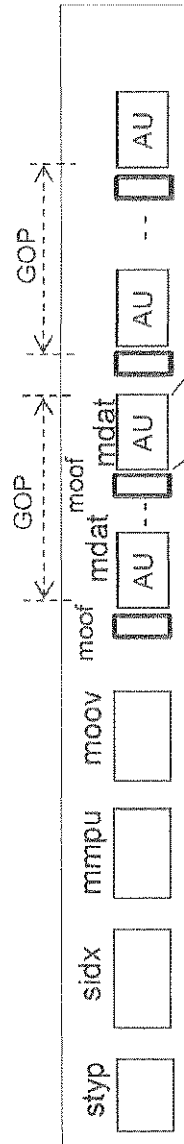


FIG.7

"Media Fragmented case" (2)

FILE



TRANSPORT PACKET

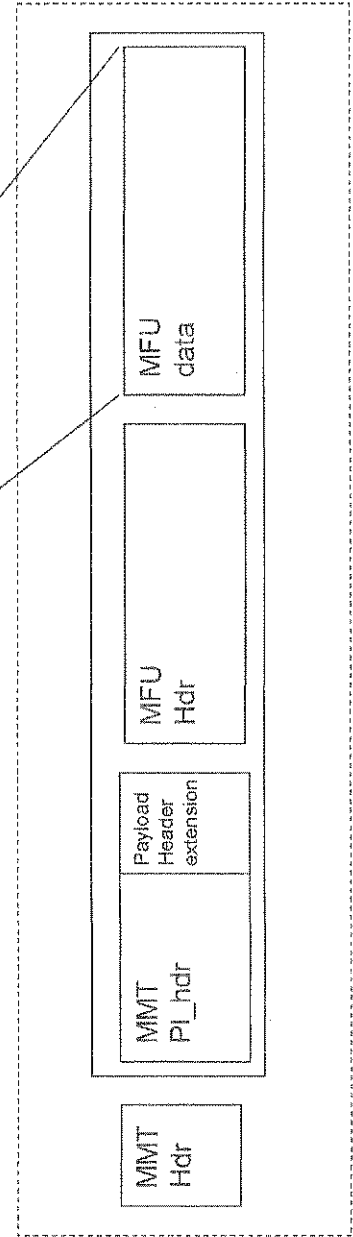


FIG.9

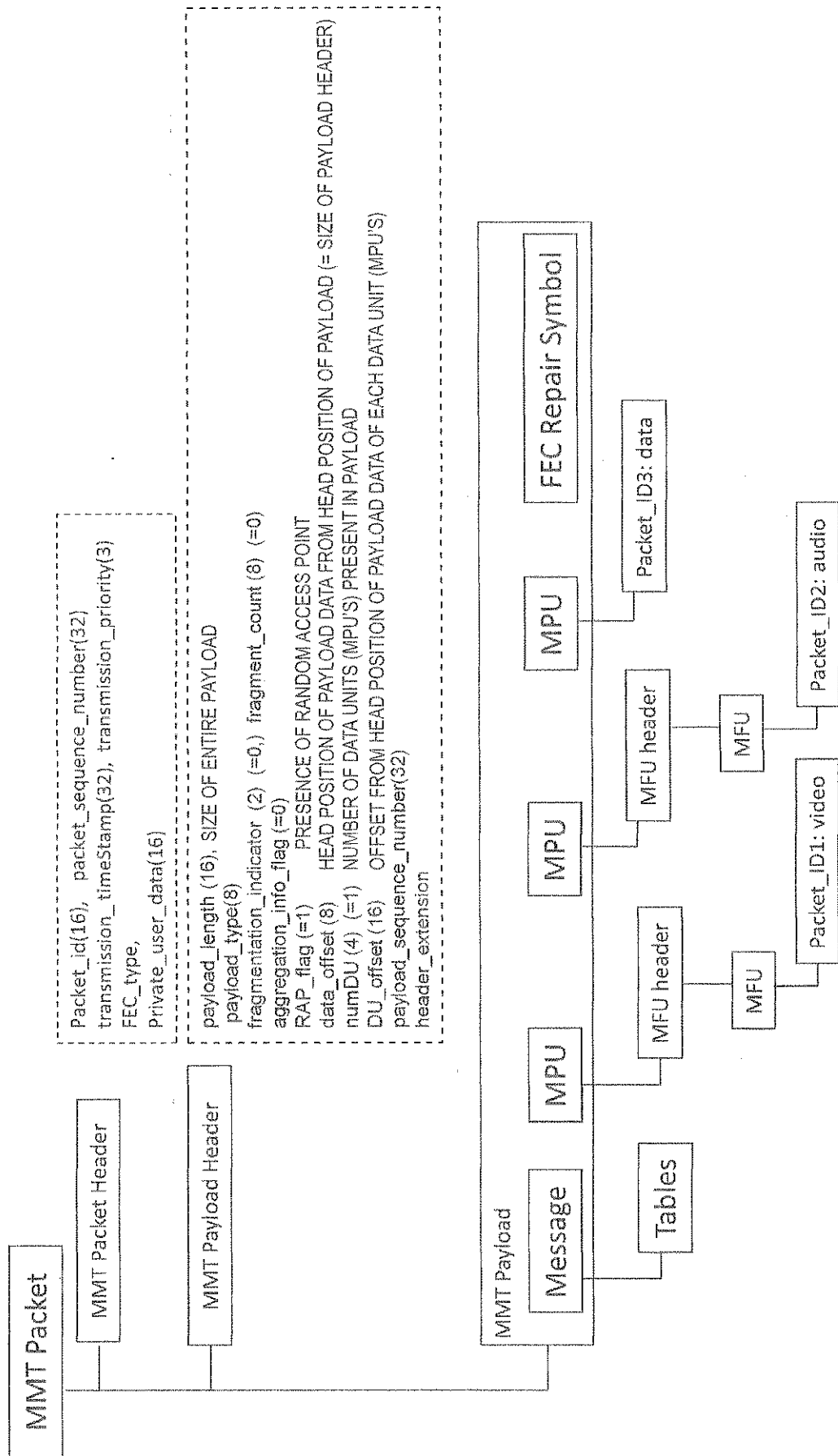


FIG.10

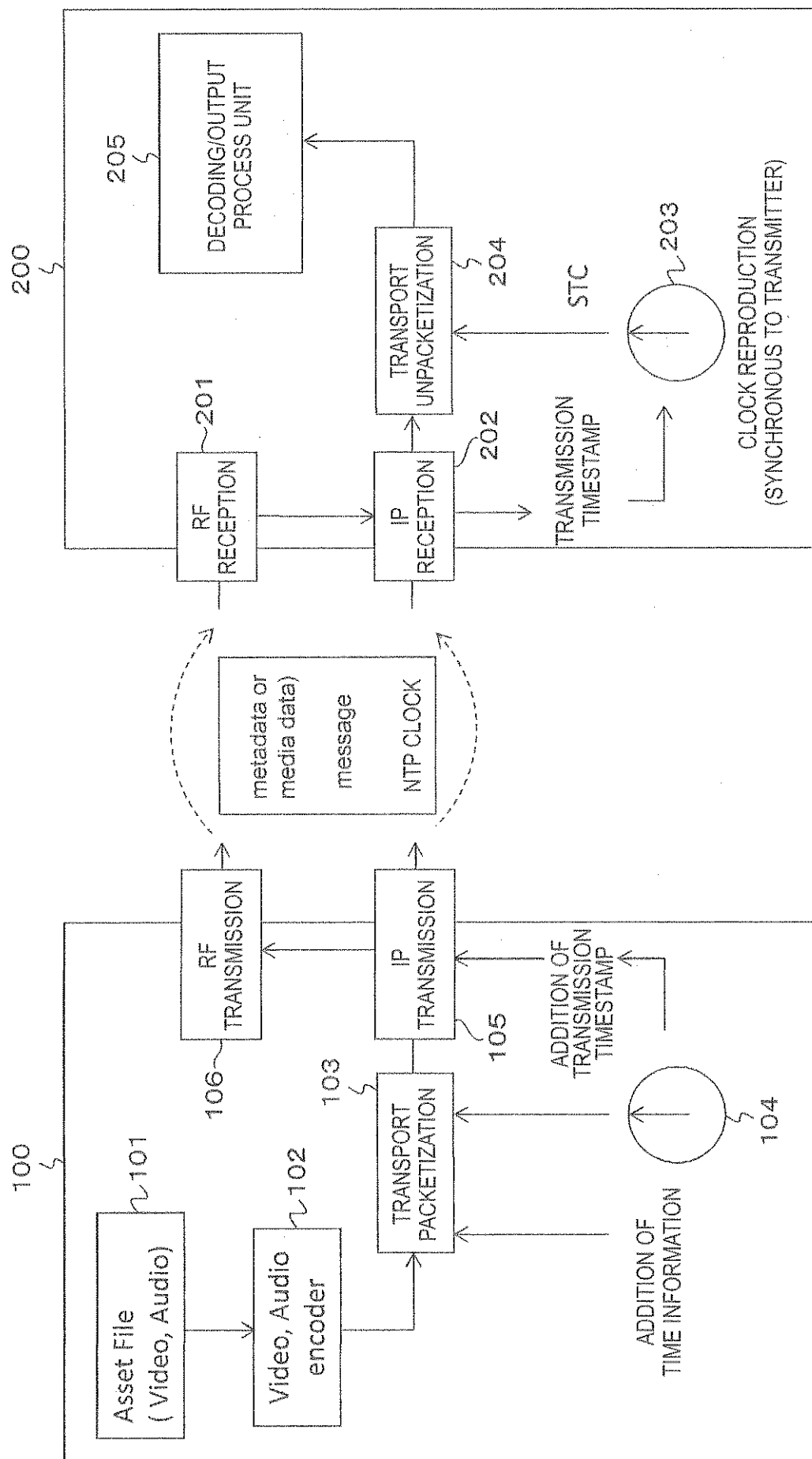


FIG.11

NTP SHORT FORMAT

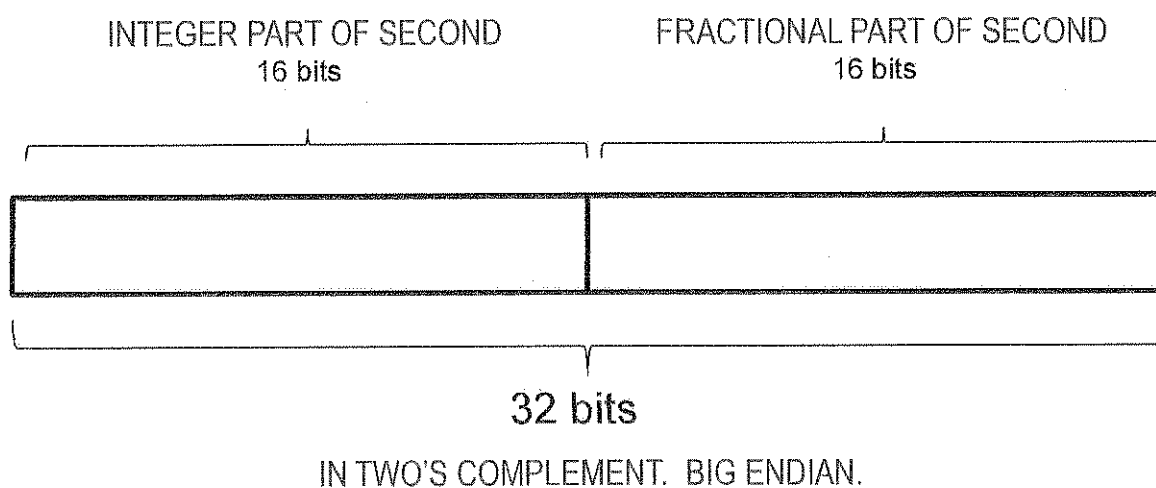
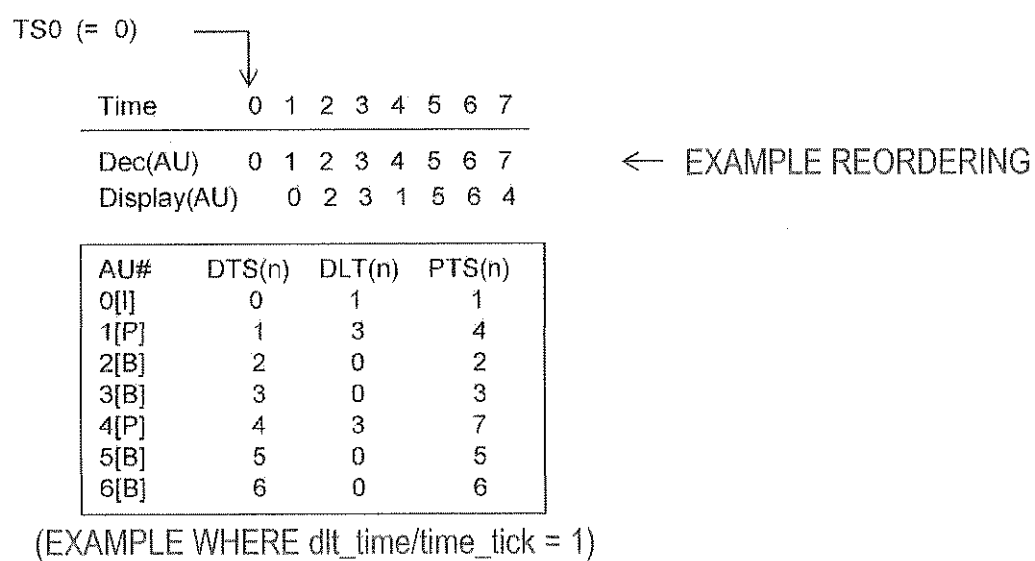
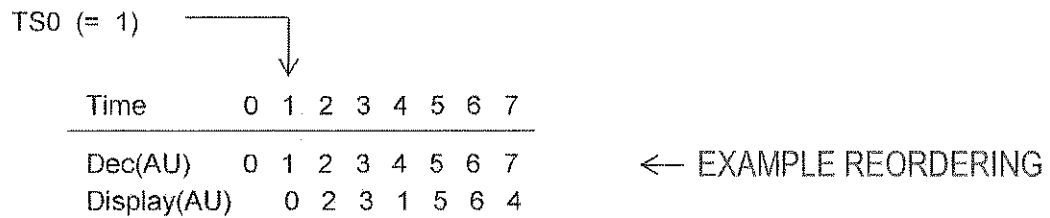
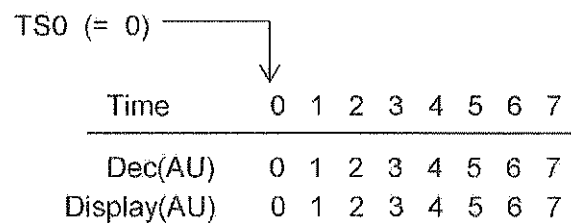
**FIG.12**

FIG.13

AU#	DTS(n)	DLT(n)	PTS(n)
0[I]	0	1	1
1[P]	1	3	4
2[B]	2	0	2
3[B]	3	0	3
4[P]	4	3	7
5[B]	5	0	5
6[B]	6	0	6

(EXAMPLE WHERE dlt_time/time_tick = 1)

FIG.14

AU#	DTS(n)	DLT(n)	PTS(n)
0	0	0	0
1	1	0	1
2	2	0	2
3	3	0	3
4	4	0	4
5	5	0	5
6	6	0	6

(EXAMPLE WHERE dlt_time/time_tick = 1)

FIG.15

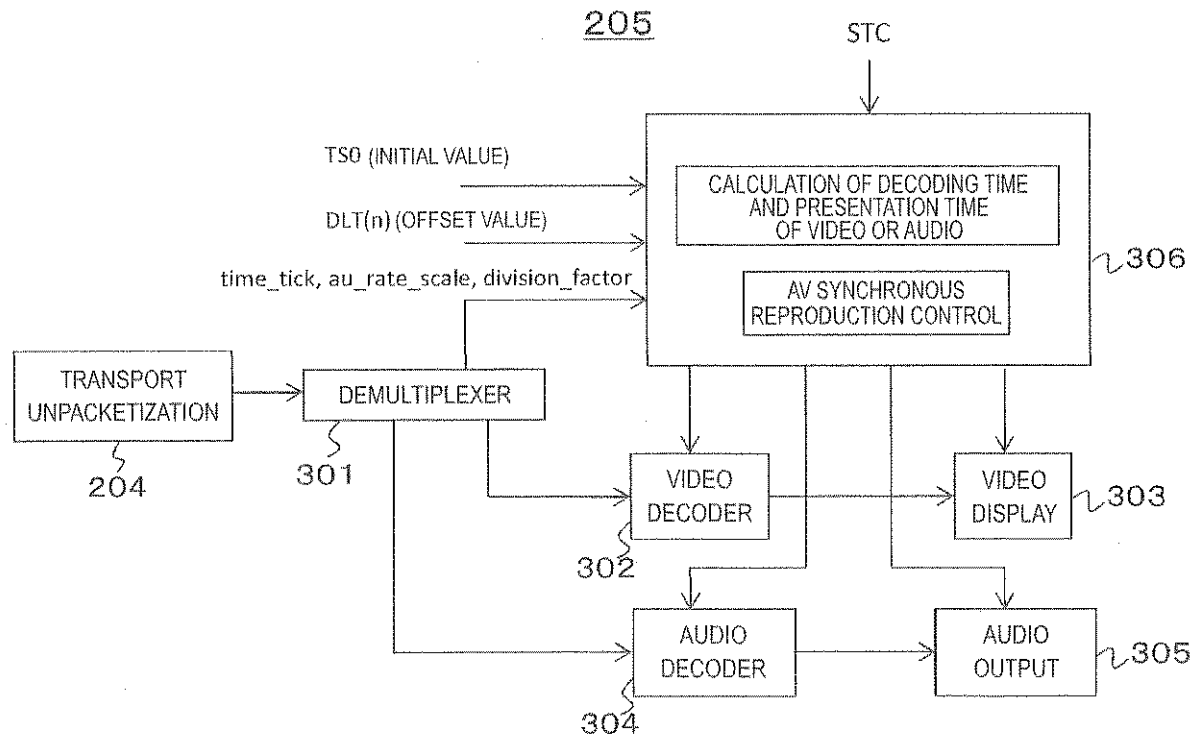


FIG.17

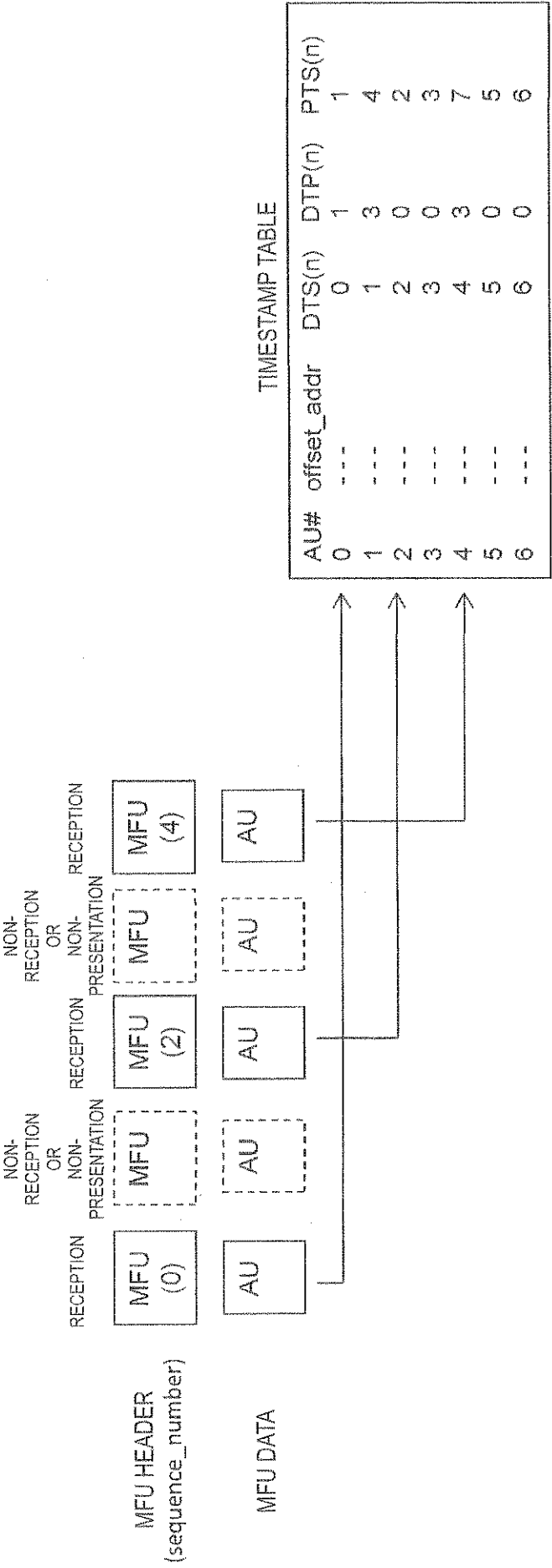
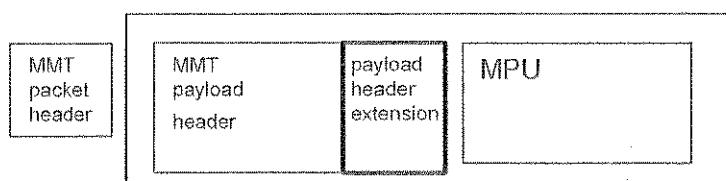
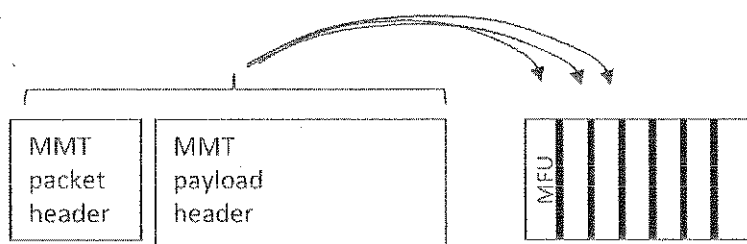


FIG.18

(a) CASE WHERE TIME INFORMATION IS TRANSMITTED IN MPU PAYLOAD HEADER



(b) CASE WHERE TIME INFORMATION IS TRANSMITTED IN MPU PAYLOAD FOR EACH AU FRAGMENT



(c) CASE WHERE TIME INFORMATION IS TRANSMITTED IN MESSAGE

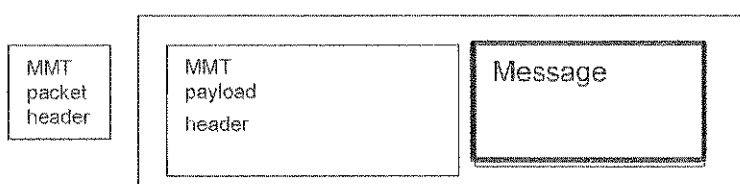


FIG.19

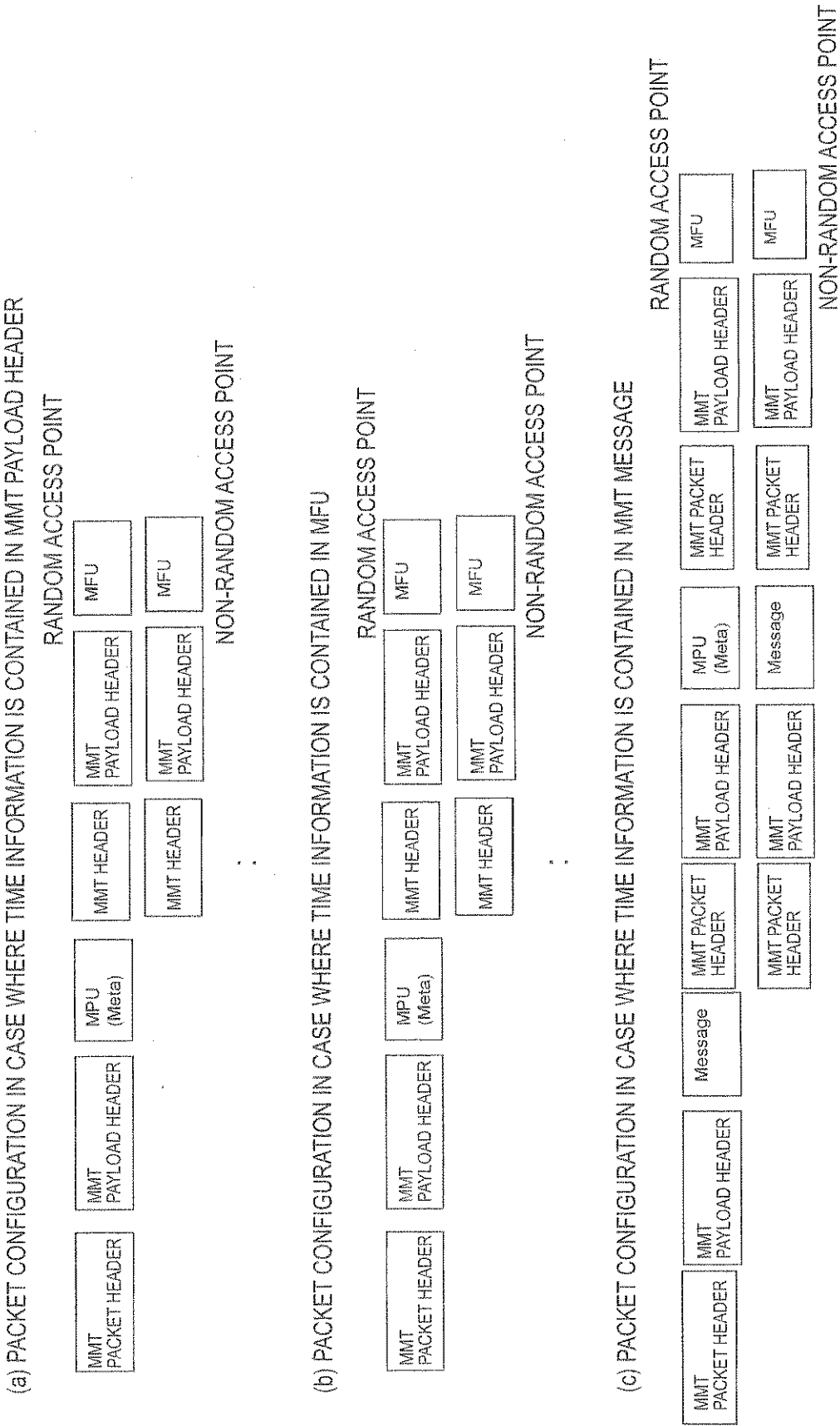


FIG.20

MMT packet overall

Syntax	No. of Bits	Format
mmt packet() {		
mmtp_header()		
mmtp_payload() {		
mmtp_payload_header()		
mmtp_payload_data()		
}		
}		

FIG.21

MMT packet header

Syntax	No. of Bits	Format
mmtp_header(){		
packet_id	16	uimsbf
packet_sequence_number	32	uimsbf
transmission_timestamp	32	uimsbf
QoS_classifier_flag	1	bslbf
flow_identifier_flag	1	bslbf
private_user_data_flag	1	bslbf
FEC_type	2	bslbf
reserved	3	bslbf
type_of_bitrate	3	bslbf
delay_sensitivity	3	bslbf
reliability_flag	1	bslbf
transmission_priority	3	bslbf
reserved	1	bslbf
flow_label	7	bslbf
extension_flag	1	bslbf
reserved	5	bslbf
private_user_data	16	bslbf
}		

FIG.22

MMT payload header

Syntax	No. of Bits	Format
mmtp_payload_header{		
payload_length	16	uimsbf
payload_type	8	bslbf
fragmentation_indicator	2	bslbf
fragment_count_flag	1	bslbf
aggregation_info_flag	1	bslbf
random_access_point_flag	1	bslbf
payload_sequence_number_flag	1	bslbf
header_extension_field_flag	1	bslbf
reserved	1	bslbf
data_offset	8	uimsbf
if (fragment_count_flag == 1)		
fragment_count	8	uimsbf
numDU	8	uimsbf
if (aggregation_info_flag == 1){		
for (l = 0 ; l < numDU ; l++){		
DU_offset	16	uimsbf
}		
if (payload_sequence_number_flag == 1)		
payload_sequence_number	32	uimsbf
if (header_extension_field_flag == 1)		
mmtp_payload_header_extension{		
}		

FIG.23

MMT payload header extension			
Syntax		No. of Bits	Format
mmtp_payload_header_extension() {			
payload_header_extension_type		16	bslbf
payload_header_extension_length		16	uimsbf
asset_type		2	bslbf
delta_sequence_type		1	bslbf
au_count_in_mpu		10	uimsbf
au_rate_scale		3	uimsbf
division_factor		2	bslbf
time_tick		2	bslbf
timestamp_type		1	bslbf
reserved		3	bslbf
timestamp_for_processing		32	uimsbf
if(delta_sequence_type) {			
for (i = 0; i < au_count_in_mpu ; i++)			
delta_variable_length_code		variable	bslbf
trailing_filler()			
} else {			
for (i = 0; i < au_count_in_mpu ; i++)			
delta_fixed_length_code		8	uimsbf
}			
}			

FIG.24

semantics

payload_header_extension_type	16bits
0x0001	providing timestamp for processing
payload_header_extension_length	16bits
asset_type	2bits
01	video
10	audio
others	reserved
time_tick	2bits
01	VALUE IN 90000-Hz PRECISION (LSB)
10	NTP shorttime
others	reserved
au_rate_scale	3bits
if(asset_type == 01) //video	
000	3750
001	3600
010	3000
011	1800
100	1500
101	900
110	750
111	reserved
else if (asset_type == 10) // audio	
000	1920
001	2089.8
010	2880
others	reserved
division_factor	2bits
if(asset_type == 01) //video	
00	1
01	1.001
others	reserved
else if (asst_type == 10) // audio	
00	1
01	2
others	reserved
timestamp_type	1bit
1	timestamp is decode_timestamp
0	timestamp is display_timestamp
timestamp_for_processing	32bits
if time_tick is "01",	
timestamp is in 90KHz precision.	
if time_tick is "10",	
timestamp is NTP short timestamp	
specified in RFC5059.	
au_count_in_mpu	10bits
delta_sequence_type	1bit
1	delta value in variable length coding
0	delta value in fixed length coding
delta_fixed_length_code	8bits
delta_variable_length_code	variable

FIG.25

MMT payload header extension

Syntax	No. of Bits	Format
mmtp_payload_header_extension() {		
payload_header_extension_type	16	bslbf
payload_header_extension_length	16	uimsbf
if (payload_header_extension_type == 0x01)		
presentation_timestamp /* in NTP format */	32	uimsbf
else if (mfu_payload_header_type == 0x02){		
decoding_timestamp /* in NTP format */	32	uimsbf
presentation_timestamp /* in NTP format */	32	uimsbf
}		
else if (mfu_payload_header_type == 0x03)		
presentation_timestamp /* in 90KHz */	32	uimsbf
else if (mfu_payload_header_type == 0x04){		
decoding_timestamp /* in 90KHz */	32	uimsbf
presentation_timestamp /* in 90KHz */	32	uimsbf
}		
else{		
}		
}		

FIG.26**semantics**

payload_header_extension_type (16bits)	
INDICATES TYPE OF MMT PAYLOAD HEADER EXTENSION	
0x01	SUPPLIES PRESENTATION TIMESTAMP IN NTP SHORT TIME FORMAT
0x02	SUPPLIES PRESENTATION TIMESTAMP AND DECODING TIMESTAMP IN NTP SHORT TIME FORMAT
0x03	SUPPLIES PRESENTATION TIMESTAMP IN 90-KHz PRECISION
0x04	SUPPLIES PRESENTATION TIMESTAMP AND DECODING TIMESTAMP IN 90-KHz PRECISION
others	reserved
presentation_timestamp (32bits)	
VALUE OF PRESENTATION TIMESTAMP	
decoding_timestamp (32bits)	
VALUE OF DECODING TIMESTAMP	

FIG.27

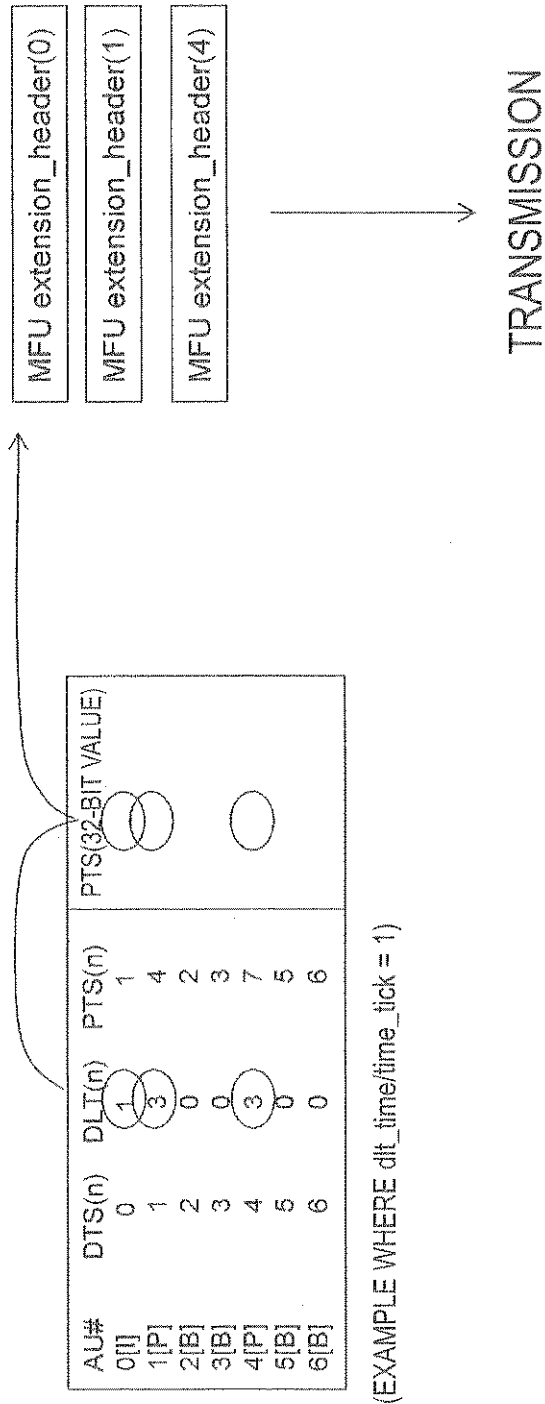


FIG.28**MFU**

Syntax	No. of Bits	Format
mfu(){		
mfu_header()		
mfu_media_data()		
}		

FIG.29**MFU header**

Syntax	No. of Bits	Format
mfu_header(){		
sequence_number	32	uimsbf
if (is_timed) {		
trackref_index	8	uimsbf
sample_number	32	uimsbf
priority	8	bslbf
dependency_counter	8	bslbf
offset	16	uimsbf
length	32	uimsbf
multiLayer_flag	1	bslbf
mfu_header_extension_flag	1	bslbf
reserved	6	bslbf
if(multilayer_flag == 1){		
dependency_id	3	bslbf
depth_flag	1	bslbf
reserved	4	bslbf
temporal_id	4	bslbf
quality_id	4	bslbf
priority_id	6	bslbf
view_id	10	bslbf

continued

FIG.30

Continued from previous page

Syntax	No. of Bits	Format
} else {		
layer_id	6	bslbf
temporal_id	3	bslbf
reserved	23	bslbf
}		
if (mfu_header_extension_flag == 1) {		
mfu_header_extension()		
}		
} else {		
item_ID	16	bslbf
}		
}		

FIG.31

Semantics

sequence_number	SEQUENCE NUMBER OF MFU IN MPU
trackref_index	MEDIA TRACK NUMBER
sample_number	ORDINAL NUMBER OF SAMPLE (ACCESS UNIT) TO WHICH RELEVANT MFU BELONGS, IN BOX moof
priority	PRIORITY LEVEL OF RELEVANT MFU IN MPU
dependency_counter	NUMBER OF MFU'S ON WHICH DECODING DEPENDS IN RELEVANT MFU
offset	OFFSET FROM BOX mdat
length	SIZE OF MFU
multiLayer_flag	PRESENCE OR ABSENCE OF MULTILAYER INFORMATION
mfu_header_extension_flag	PRESENCE OR ABSENCE OF mfu_extension
dependency_id	ID'S INDICATING VARIOUS DEPENDENCE RELATIONSHIPS BETWEEN MFU'S
depth_flag	
temporal_id	
quality_id	
priority_id	
view_id	ID'S INDICATING VARIOUS DEPENDENCE RELATIONSHIPS BETWEEN MFU'S
layer_id	
Item_ID	ID OF FILE

FIG.32

Syntax	No. of Bits	Format
mfu_header_extension(){		
mfu_header_extension_type	8	bslbf
mfu_extension_length	8	uimsbf
asset_type	2	bslbf
au_rate_scale	3	uimsbf
division_factor	2	bslbf
time_tick	2	bslbf
timestamp_type	1	bslbf
reserved	6	bslbf
timestamp_for_processing	32	uimsbf
delta_fixed_length_code	8	uimsbf
}		

FIG.33

semantics

asset_type	2bits
01	video
10	audio
others	reserved
Time tick	2bits
01	90000 Hz
10	NTP shorttime
others	reserved
AU_rate_scale 3bits	
if(asset_type == 01) //video	
000	3750
001	3600
010	3000
011	1800
100	1500
101	900
110	750
111	reserved
else if (asset_type == 10) // audio	
000	1920
001	2089.8
010	2880
others	reserved
others	reserved

division_factor	2bits
if(asset_type == 01) //video	
00	1
01	1.001
others	reserved
else if (asst_type == 10) // audio	
00	1
01	2
timestamp_type	1bit
1	timestamp is decode_timestamp
0	timestamp is display_timestamp
timestamp_for_processing	32bits
if time_tick is "01",	
timestamp is in 90KHz precision.	
if time_tick is "10",	
timestamp is NTP short timestamp	
specified in RFC5059.	
delta_fixed_length_code	8bits

FIG.34

MFU header

Syntax	No. of Bits	Format
mfu_header{		
sequence_number	32	uimsbf
if (is_timed) {		
trackref_index	8	bslbf
sample_number	32	uimsbf
priority	8	bslbf
dependency_counter	8	bslbf
offset	16	uimsbf
length	32	uimsbf
multiLayer_flag	1	bslbf
timestamp_type	2	bslbf
time_tick	1	bslbf
reserved	4	bslbf
if(multilayer_flag == 1){		
dependency_id	3	bslbf
depth_flag	1	bslbf
reserved	4	bslbf
temporal_id	4	bslbf
quality_id	4	bslbf
priority_id	6	bslbf
view_id	10	bslbf

continued

FIG.35

Continued from previous page

Syntax	No. of Bits	Format
} else {		
layer_id	6	bslbf
temporal_id	3	bslbf
reserved	23	bslbf
}		
} else {		
item_ID	16	bslbf
}		
}		

FIG.36

semantics

Timestamp_type (2bits)	
INDICATES WHETHER OR NOT TIMESTAMP IS SUPPLIED AT HEAD OF MFU PAYLOAD	
00	TIMESTAMP IS NOT SUPPLIED. 1ST BYTE OF MFU PAYLOAD IS EQUAL TO HEAD BYTE OF FRAGMENTED MPU PAYLOAD
01	PRESENTATION TIMESTAMP IS SUPPLIED AT HEAD OF MFU PAYLOAD. HEAD BYTE OF FRAGMENTED MPU PAYLOAD IS SUPPLIED IMMEDIATELY AFTER PRESENTATION TIMESTAMP
10	DECODING TIMESTAMP AND PRESENTATION TIMESTAMP ARE BOTH SUPPLIED AT HEAD OF MFU PAYLOAD. HEAD BYTE OF FRAGMENTED MPU PAYLOAD IS SUPPLIED IMMEDIATELY AFTER TIMESTAMPS
11	reserved
Time tick (1bit)	
DESCRIBES PRECISION OF VALUE OF TIMESTAMP	
0	90000 Hz PRECISION
1	NTP short time FORMAT

FIG.37

Syntax	No. of Bits	Format
mfu_payload(){		
if(timestamp_type == 01)		
presentation_timestamp	32	uimsbf
else if(timestamp_type == 10){		
decoding_timestamp	32	uimsbf
presentation_timestamp	32	uimsbf
}		
else{		
}		
payload_data		
}		

semantics

presentation_timestamp (32bits)
VALUE OF PRESENTATION TIMESTAMP
decoding_timestamp (32bits)
VALUE OF DECODING TIMESTAMP

FIG.38

Syntax	No. of Bits	Format
Mfu_header(){		
sequence_number	32	uimsbf
if (is_timed) {		
trackref_index	8	bslbf
sample_number	32	uimsbf
priority	8	bslbf
dependency_counter	8	bslbf
offset	16	uimsbf
length	32	uimsbf
multiLayer_flag	1	bslbf
payload_header_flag	1	bslbf
reserved	6	bslbf
if(multiLayer_flag == 1){		
dependency_id	3	bslbf
depth_flag	1	bslbf
reserved	4	bslbf
temporal_id	4	bslbf
quality_id	4	bslbf
priority_id	6	bslbf
view_id	10	bslbf

Continued to next page

FIG.39

Continued from previous page

Syntax	No. of Bits	Format
} else {		
layer_id	6	bslbf
temporal_id	3	bslbf
reserved	23	bslbf
}		
} else {		
item_ID	16	bslbf
}		
}		

FIG.40

semantics

Payload_header_flag (1bit)

INDICATES WHETHER OR NOT MFU PAYLOAD HEADER IS PRESENT AT HEAD OF MFU PAYLOAD

0 MFU PAYLOAD HEADER IS NOT PRESENT. 1ST BYTE OF MFU PAYLOAD IS
EQUAL TO HEAD BYTE OF FRAGMENTED MPU PAYLOAD

1 MFU PAYLOAD HEADER IS PRESENT. HEAD BYTE OF FRAGMENTED MPU PAYLOAD IS
SUPPLIED IMMEDIATELY AFTER MFU PAYLOAD HEADER

FIG.41

Syntax	No. of Bits	Format
mfu_payload(){		
if (payload_header_flag == 1){		
mfu_payload_header_type	8	bsbf
mfu_payload_header_length	8	uimsbf
if (mfu_payload_header_type == 0x01)		
presentation_timestamp /* in NTP format */	32	uimsbf
else if (mfu_payload_header_type == 0x02){		
decoding_timestamp /* in NTP format */	32	uimsbf
presentation_timestamp /* in NTP format */	32	uimsbf
}		
else if (mfu_payload_header_type == 0x03)		
presentation_timestamp /* in 90KHz */	32	uimsbf
else if (mfu_payload_header_type == 0x04){		
decoding_timestamp /* in 90KHz */	32	uimsbf
presentation_timestamp /* in 90KHz */	32	uimsbf
}		
else{		
}		
}		
payload_data		
}		

FIG.42

semantics

mfu_payload_header_type (8bits)	INDICATES TYPE OF MFU PAYLOAD HEADER
0x01	SUPPLIES PRESENTATION TIMESTAMP IN NTP SHORT TIME FORMAT
0x02	SUPPLIES PRESENTATION TIMESTAMP AND DECODING TIMESTAMP IN NTP SHORT TIME FORMAT
0x03	SUPPLIES PRESENTATION TIMESTAMP IN 90-KHZ PRECISION
0x04	SUPPLIES PRESENTATION TIMESTAMP AND DECODING TIMESTAMP IN 90-KHZ PRECISION
others	reserved
mfu_payload_header_length (8bits)	INDICATES SIZE OF mfu PAYLOAD HEADER AFTER THIS ELEMENT IN NUMBER OF BYTES
presentation_timestamp (32bits)	VALUE OF PRESENTATION TIMESTAMP
decoding_timestamp (32bits)	VALUE OF DECODING TIMESTAMP

FIG.43

Timestamp Message

Syntax	No. of Bits	Format
timestamp_message() {		
message_id	16	
version	8	
length	16	
extension {		
timestamp_table()		
}		
}		

FIG.44

Syntax	No. of Bits	Format
Timestamp_table() {		
table_id	8	uimsbf
version	8	uimsbf
length	16	uimsbf
packet_id	16	uimsbf
asset_type	2	bslbf
delta_sequence_type	1	bslbf
au_count_in_mpu	10	uimsbf
au_rate_scale	3	uimsbf
division_factor	2	bslbf
time_tick	2	bslbf
timestamp_type	1	bslbf
reserved	3	bslbf
timestamp_for_processing	32	uimsbf
if(delta_sequence_type) {		
for (i = 0; i < au_count_in_mpu ; i++)		
delta_variable_length_code	variable	
trailing_filler()		
} else{		
for (i = 0; i < au_count_in_mpu ; i++)		
delta_fixed_length_code	8	uimsbf
}		
}		

FIG.45

Package Access Message

Syntax	No. of Bits	Format
Package_Access_message() {		
message_id	16	uimsbf
version	8	uimsbf
length	32	uimsbf
extension {		
number_of_tables ; N1	8	uimsbf
for (i=0; i < N1; i++) {		
table_id	8	uimsbf
table_version	8	uimsbf
table_length	16	uimsbf
}		
message_payload {		
for (i=0; i < N1; i++) {		
table()		
}		
}		

FIG.46

Time	0	1	2	3	4	5	6	7
Dec(AU)	D(0)	D(1)	D(2)	D(3)	D(4)	D(5)	D(6)	D(7)
Display(AU)		R(0)	R(2)	R(3)	R(1)	R(5)	R(6)	R(4)
DLT(AU)	1	3	0	0	3	0	0	3

FIG.47

VARIABLE LENGTH CODE TABLE

CODING INPUT VALUE	VARIABLE LENGTH CODE	CODE LENGTH (BITS)
0 ::	1	1
1 ::	010	3
2 ::	011	3
3 ::	00100	5
4 ::	00101	5
5 ::	00110	5
6 ::	00111	5
7 ::	0001000	7
8 ::	0001001	7
9 ::	0001010	7
10 ::	0001011	7
11 ::	0001100	7
12 ::	0001101	7
13 ::	0001110	7
14 ::	0001111	7
15 ::	000010000	9

NUMBER OF SUCCESSIVE "0"S INDICATES CODE LENGTH.

CODE LENGTH IS 1 IF THERE IS NO "0" AND 3 IF THERE IS ONE "0."

CODE LENGTH IS 5 IF THERE ARE TWO "0"S AND 7 IF THERE ARE THREE "0"S.

FIG.48

ESTIMATION OF CODE LENGTH

GOP length (M)	TOTAL BIT LENGTH	FIXED 4-BIT LENGTH
M = 3 (IPBB) IBBP B_probability: 2/4	3 + 5 + 2 = 10 (3+7) bits	16 bits
M = 4 (IPBBB) IBBBP B_probability: 3/5	3 + 5 + 3 = 11 (3+8) bits	20bits
M = 5 (IPBBBB) IBBBBP B_probability: 4/6	3 + 5 + 4 = 12 (3+9) bits	24bits
M = 6 (IPBBBBB) IBBBBBP B_probability: 5/7	3 + 5 + 5 = 13 (3+10) bits	28bits
M = 7 (IPBBBBBB) IBBBBBBP B_probability: 6/8	3 + 7 + 6 = 16 (3+13) bits	32bits
M = 8 (IPBBBBBBB) IBBBBBBBP B_probability: 7/9	3 + 7 + 7 = 17 (3+14) bits	36bits
M = 15 (IPBBBBBBBBBBBBBBBB) IBBBBBBBBBBBBBBBBBP B_probability: 14/16	3 + 9 + 14 = 26 (3+23) bits	64bits