

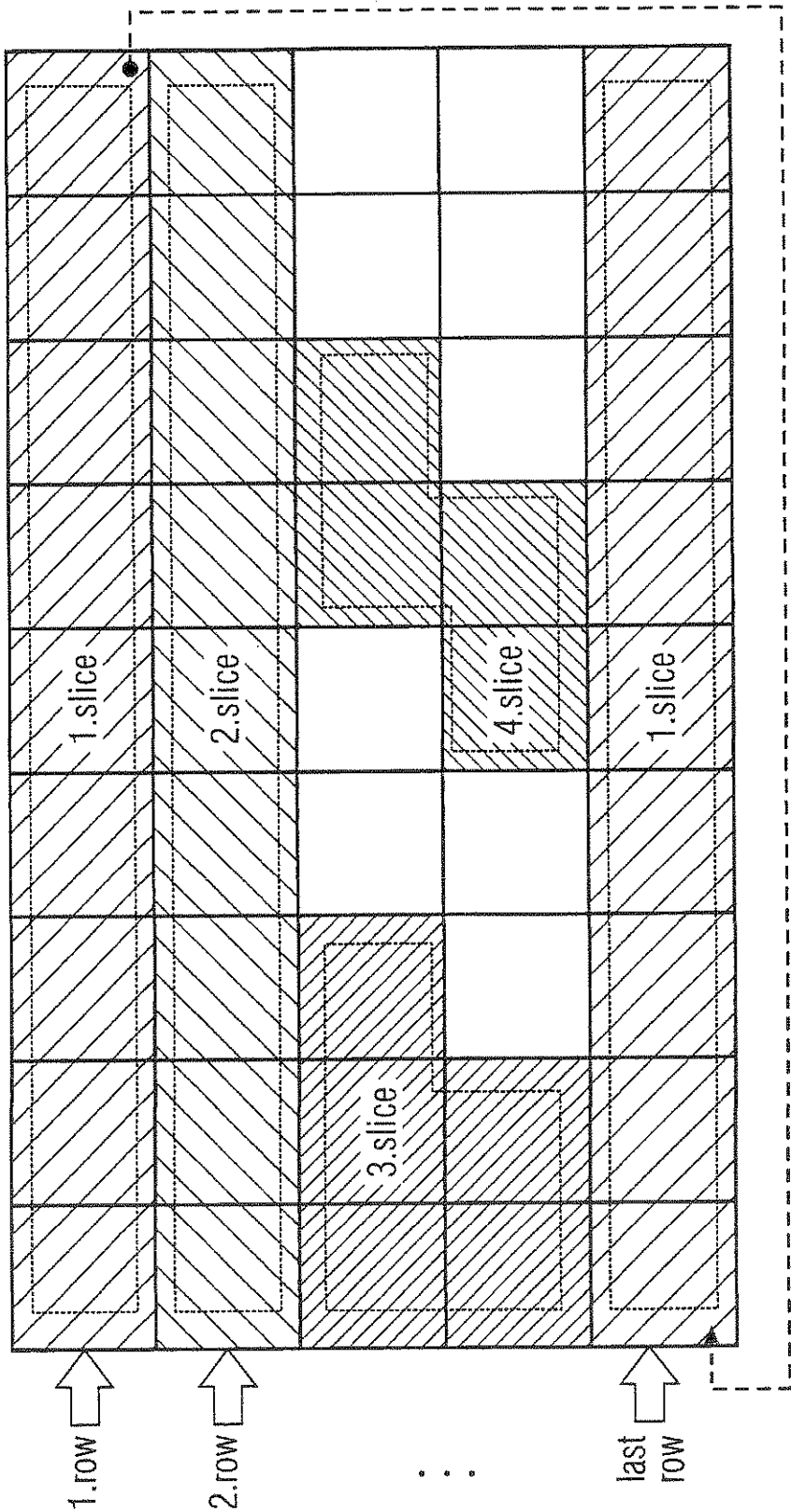


FIG 1

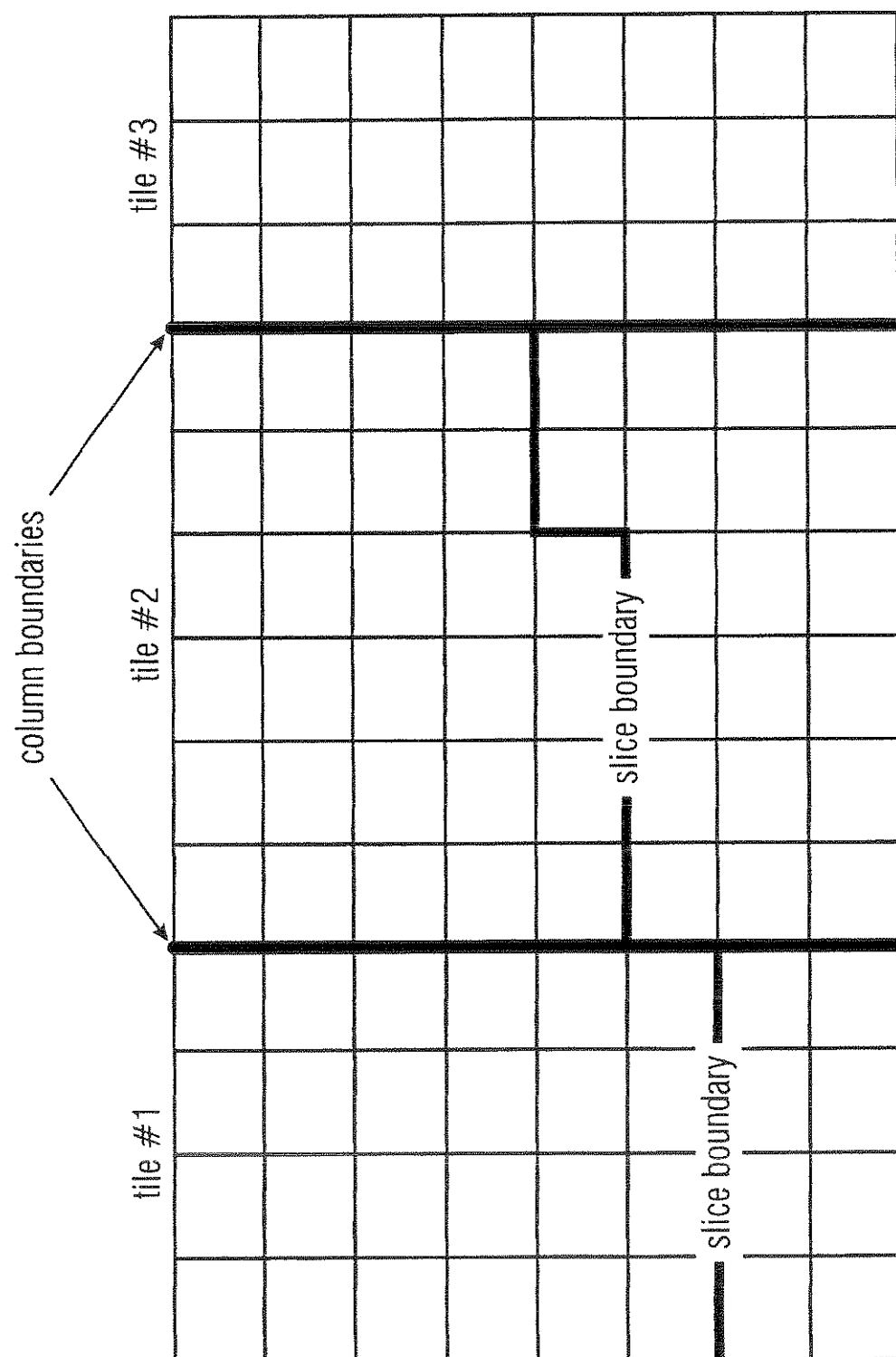


FIG 2

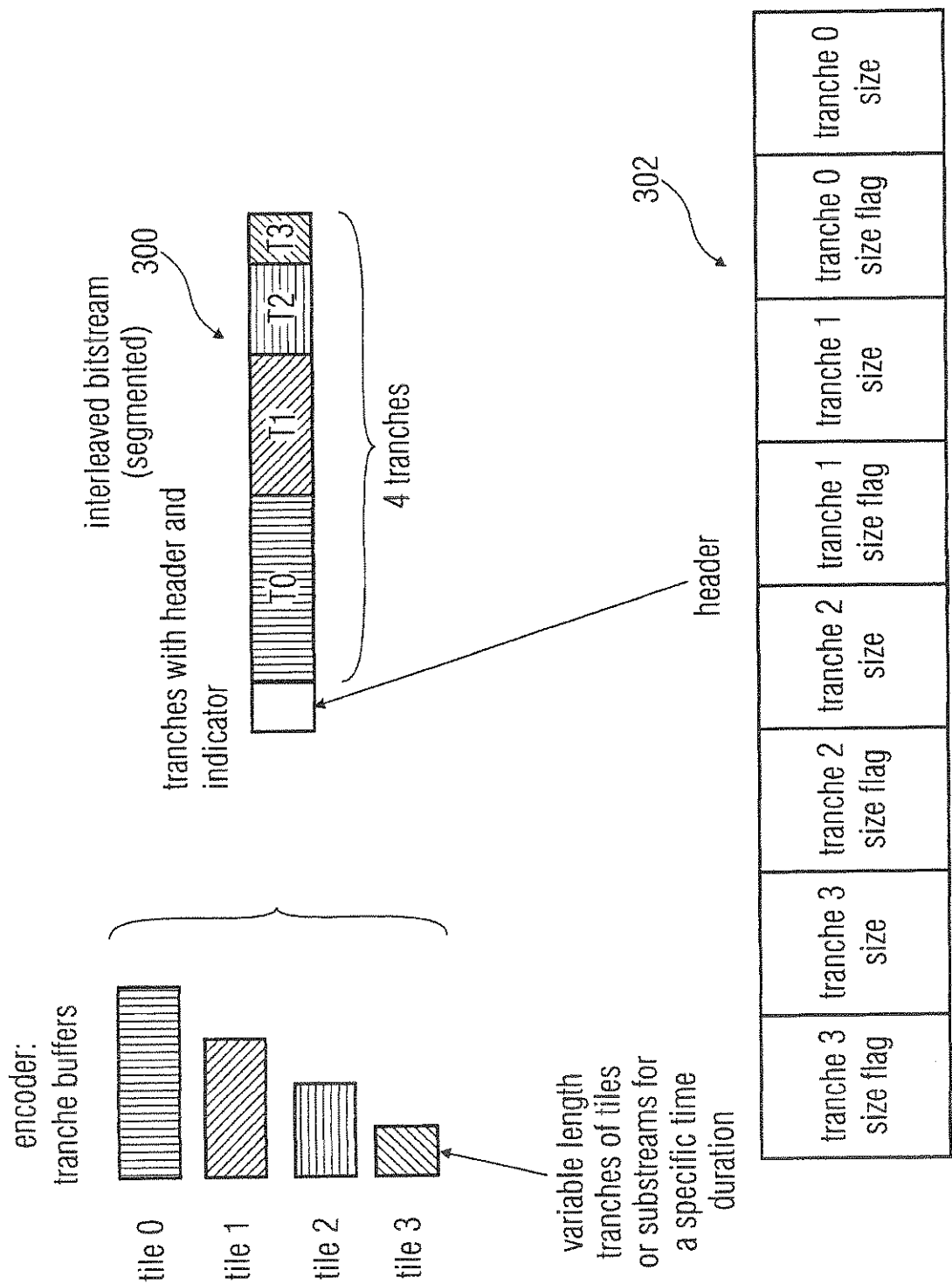
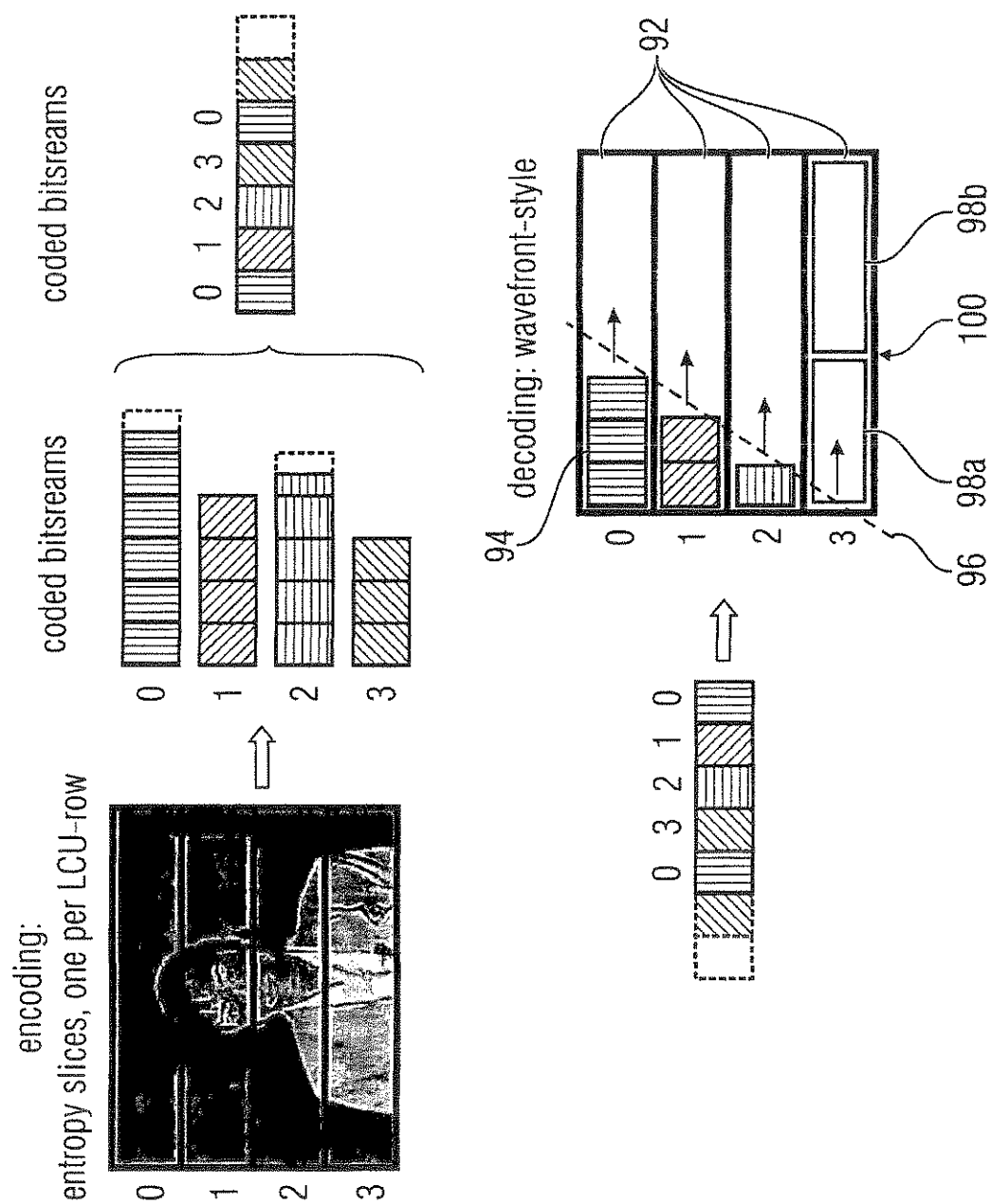


FIG 3



45E

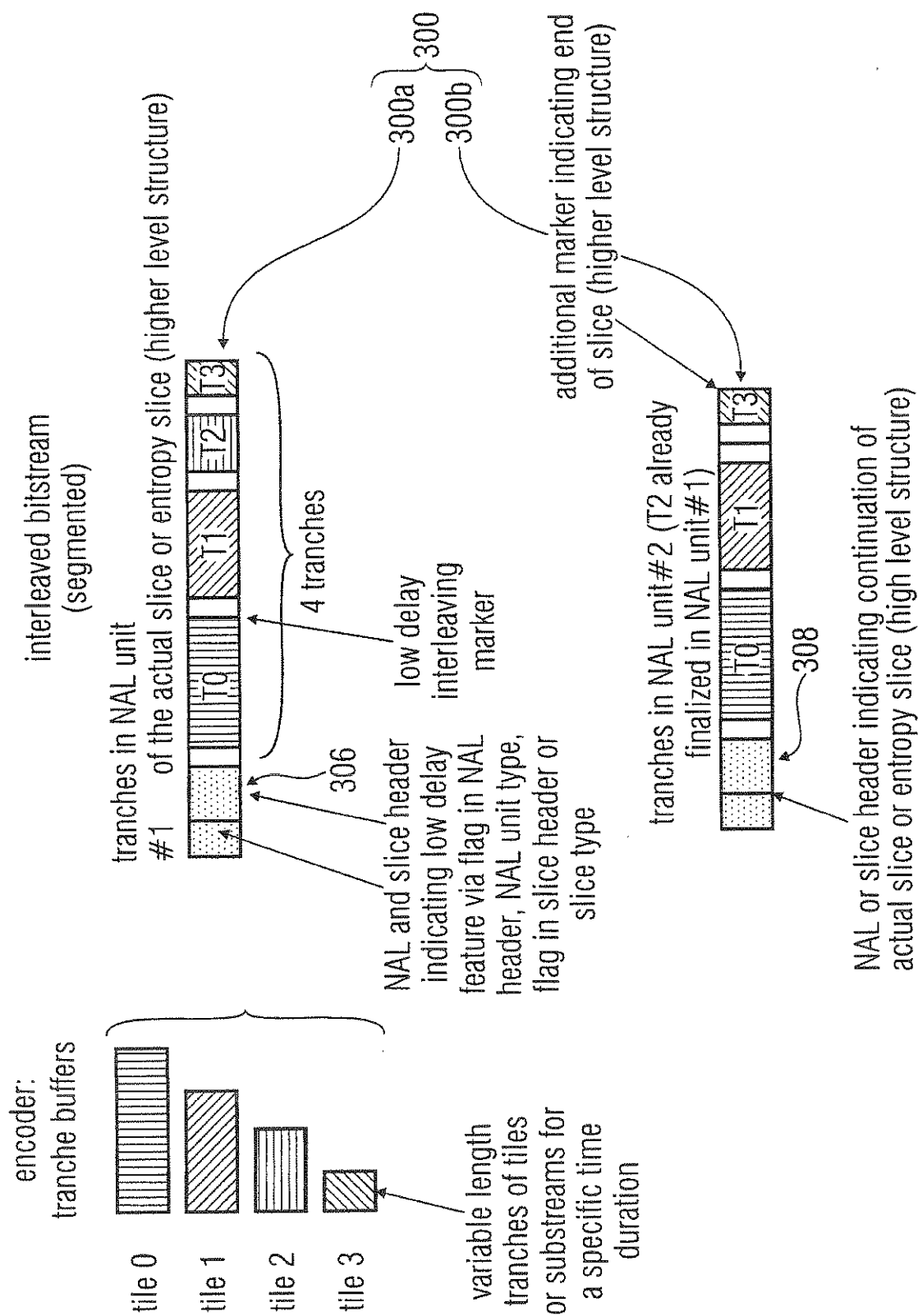


FIG 5

6/17

	Descriptor
nal_unit(NumBytesInNALunit, num_low_delay_tranches) {	
forbidden_zero_bit	f(1)
nal_ref_flag	u(1)
nal_unit_type	u(6)
NumBytesInRBSP=0	
nalUnitHeaderBytes=1	
if(nal_unit_type==1 nal_unit_type==4 nal_unit_type==5) {	
temporal_id	u(3)
output_flag	u(1)
low_delay_encapsulation_flag	u(1)
low_delay_cyclic_flag	u(1)
reserved_one_2bits	u(2)
nalUnitHeaderBytes += 1	
}	
if(low_delay_encapsulation_flag != 1)	
{	
for(i=nalUnitHeaderBytes; i<NumBytesInNALunit; i++) {	
if(i+2<NumBytesInNALunit && next_bits(24) == 0x000003) {	
rbsp_byte[NumBytesInRBSP++]	b(8)
rbsp_byte[NumBytesInRBSP++]	b(8)
i += 2	
emulation_prevention_three_byte /* equal to 0x03 */	f(8)
} else	
rbsp_byte[NumBytesInRBSP++]	b(8)
}	
}	
else{	
for(i=0, i++, i<num_low_delay_tranches){	
NumLDBytesInRBSP[i]=0	
}	

FIG 6

FIG 6A

FIG 6B

FIG 6A

7/17

<pre> tranche_id=-1 for (i=0, i++, i<num_low_delay_tranches){ NumBytesInRBSP[i]=0 } for(i=nalUnitHeaderBytes;i<NumBytesInNALUnit;i++) { if(i+2<NumBytesInNALUnit && next_bits(24) == 0x000002) { if(low_delay_cyclic_flag) { tranche_id=(tranche_id++) % num_low_delay_tranches i+=2 } else{ tranche_id i+=3 } } else if(i+2<NumBytesInNALUnit && next_bits(24) == 0x000003) { LD_rbsp_byte[NumLDBytesInRBSP[tranche_id]++] [tranche_id] LD_rbsp_byte[NumLDBytesInRBSP[tranche_id]++] [tranche_id] i+=2 emulation_prevention_three_byte /* equal to 0x03 */ } else LD_rbsp_byte[NumLDBytesInRBSP[tranche_id]++] [tranche_id] } for (i=0, i++, i<num_low_delay_tranches){ for (i=0, i++, i< NumLDBytesInRBSP[i]){ rbsp_byte[NumBytesInRBSP++]=LD_rbsp_byte[i][i] } } } </pre>	<p>u(8)</p> <p>b(8)</p> <p>b(8)</p> <p>f(8)</p> <p>b(8)</p>
--	---

FIG 6B

FIG 6

FIG 6A

FIG 6B

8/17

...	
low_delay_encapsulaiton_present_flag	u(1)
if(low_delay_encapsulaiton_present_flag ==1){	
low_delay_cyclic_flag	u(1)
num_low_delay_tranches_flag	u(1)
if (num_low_delay_tranches_flag==1){	
num_low_delay_tranches	ue(v)
for (i=0; i < num_low_delay_tranches; i++){	
low_delay_tranche_lenght_minus1 [i]	ue(v)
}	
}	
}	
...	

FIG 7

low_delay_slice_layer_rbsp() {	Descriptor
slice_header()	
tranche_id=0	
for (i=0, i++ , i<num_low_delay_tranches){	
tranche_slice_data()	
}	
rbps_slice_trailing_bits()	

FIG 8

9/17

	Descriptor
slice_header() {	
entropy_slice_flag	u(1)
low_delay_slice_flag	u(1)
....	
first_slice_in_pic_flag	u(1)
if(first_slice_in_pic_flag==0){	
slice_adress	u(v)
if(low_delay_slice_flag==1)	
no_cabac_reset_flag	u(1)
}	
....	
if(slice_type==P slice_type==B)	
5_minus_max_num_merge_cand	ue(v)
for(i=0; i<num_substreams_minus1+1; i++){	
substream_length_mode	u(2)
substream_length[i]	u(v)
}	
}	
}	

FIG 9

	Descriptor
substream_marker(gradual_decoder_refresh_steps) {	
forbidden_zero_bit	f(1)
nal_ref_flag	u(1)
nal_unit_type	u(6)
substream_ID	u(8)
if(gradual_decoder_refresh_steps< 1)	
is_intra	u(1)
reserved_one_7bits	u(7)
}	
}	

FIG 10

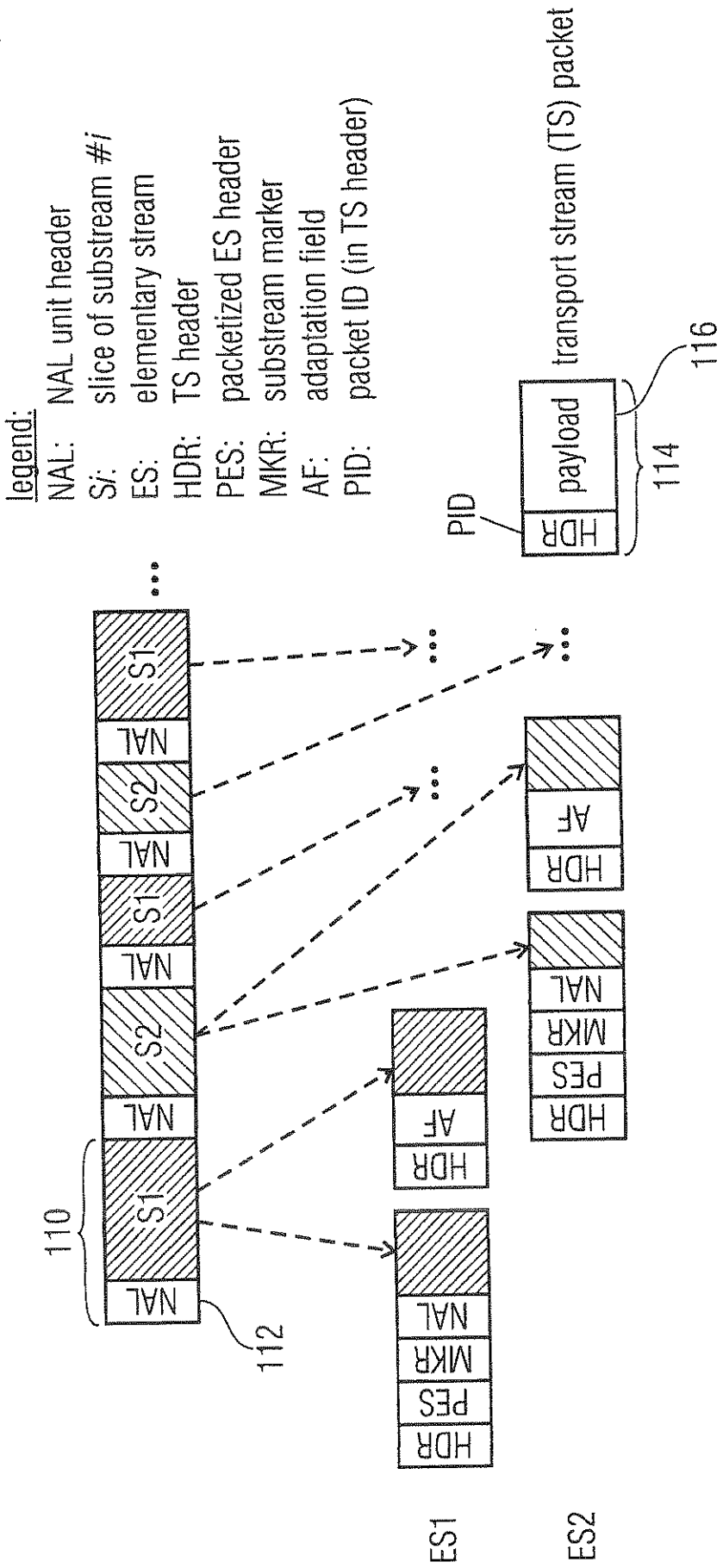


FIG 11

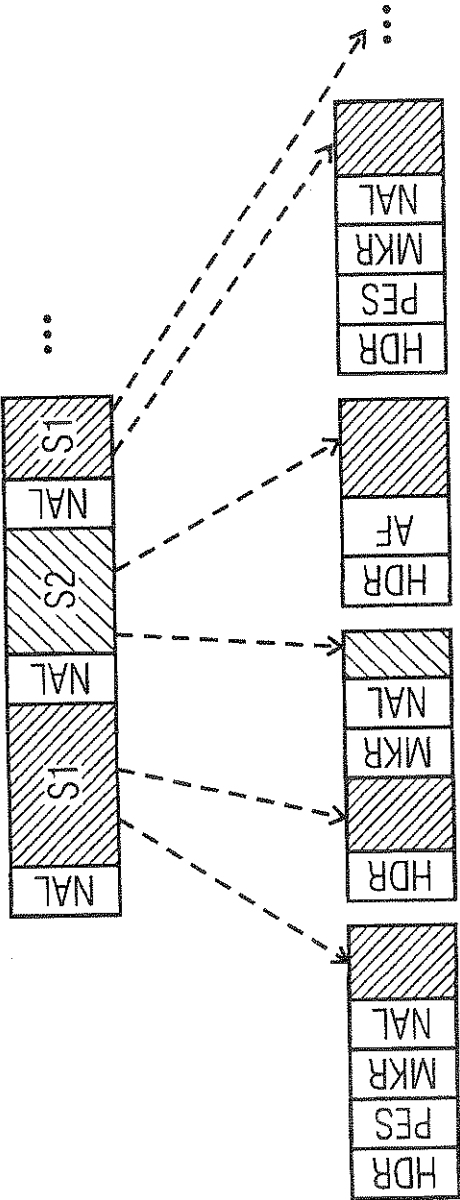


FIG 12

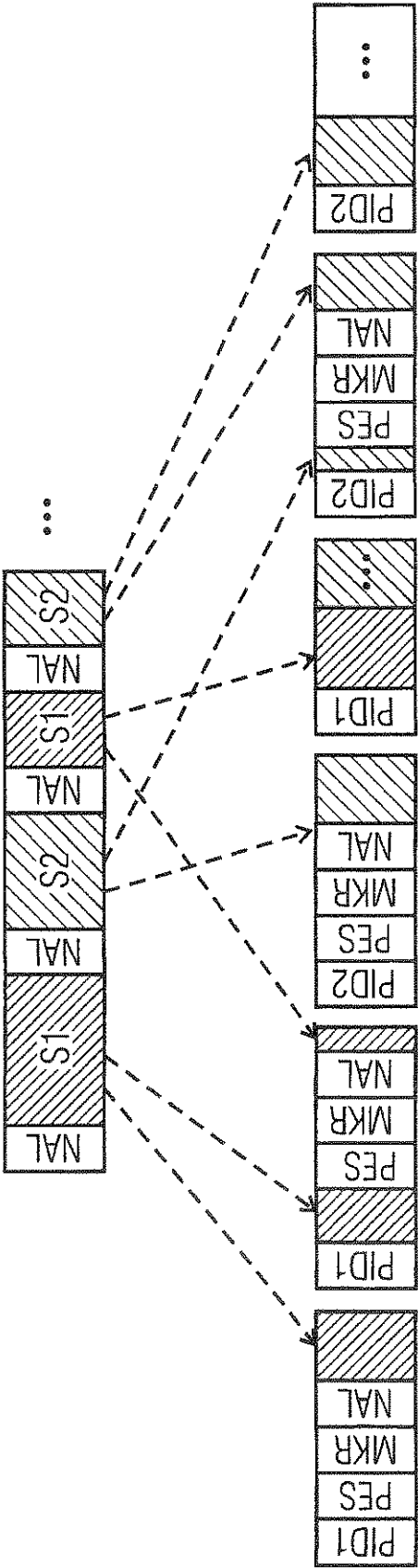


FIG 13

13/17

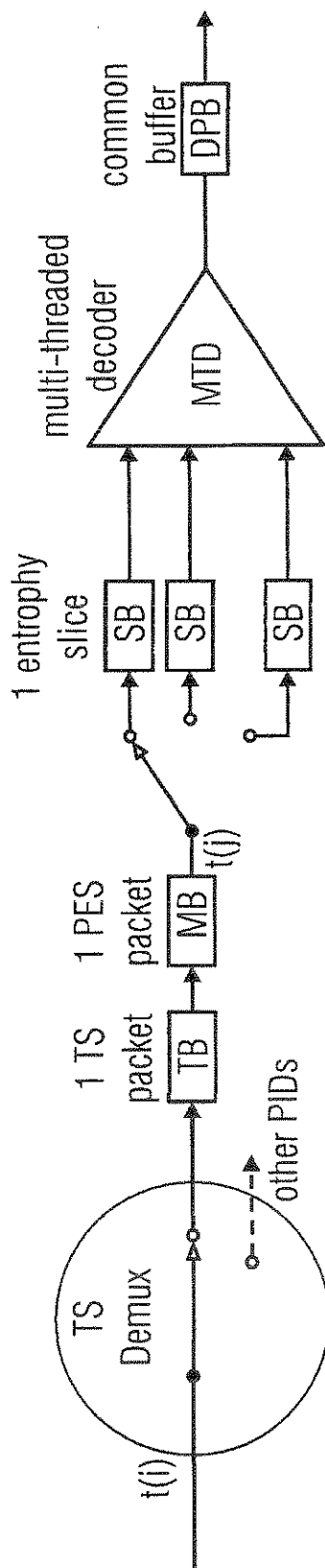


FIG 14

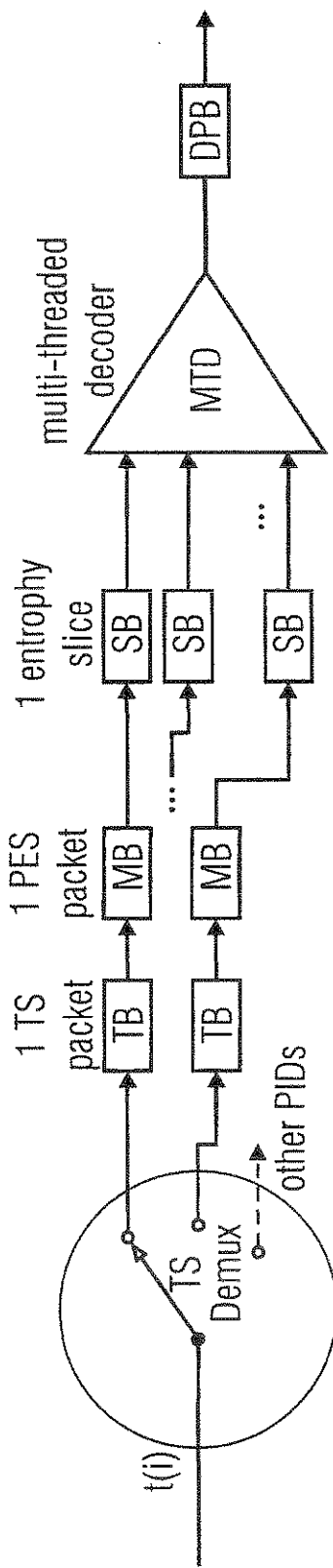


FIG 15

15/17

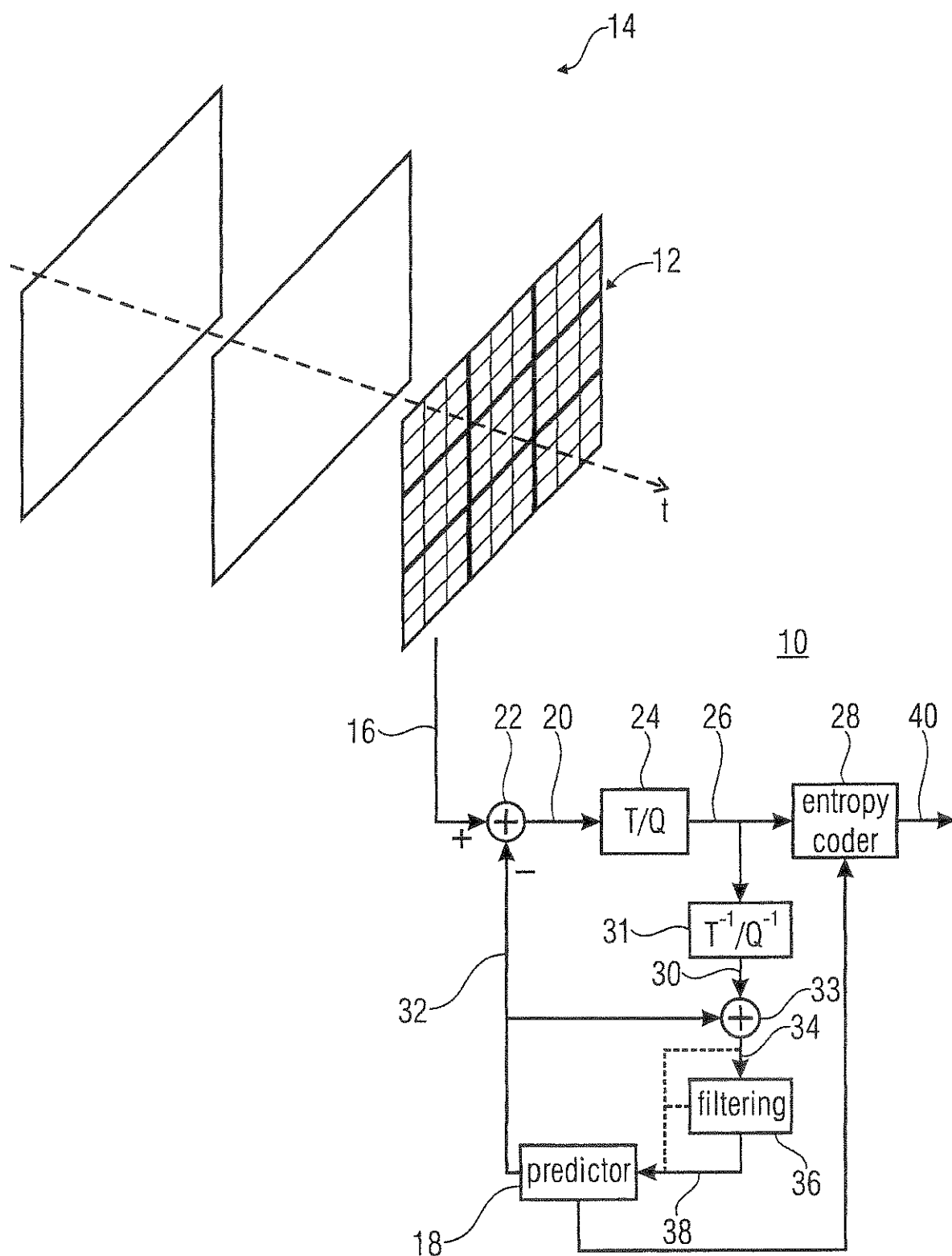


FIG 16

16/17

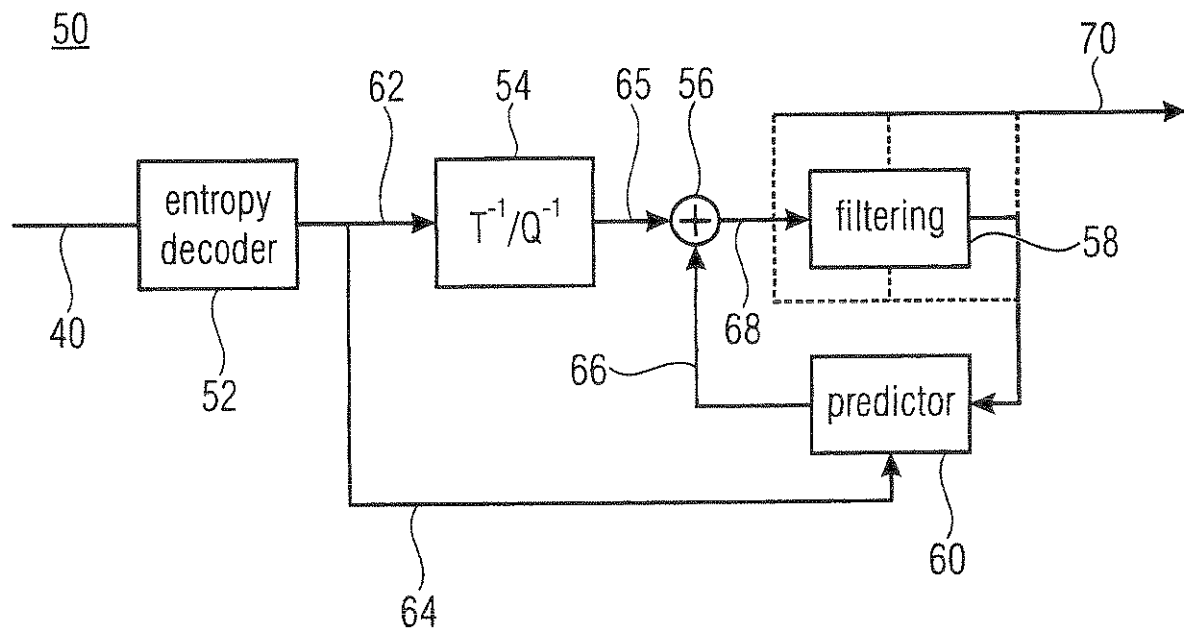


FIG 17

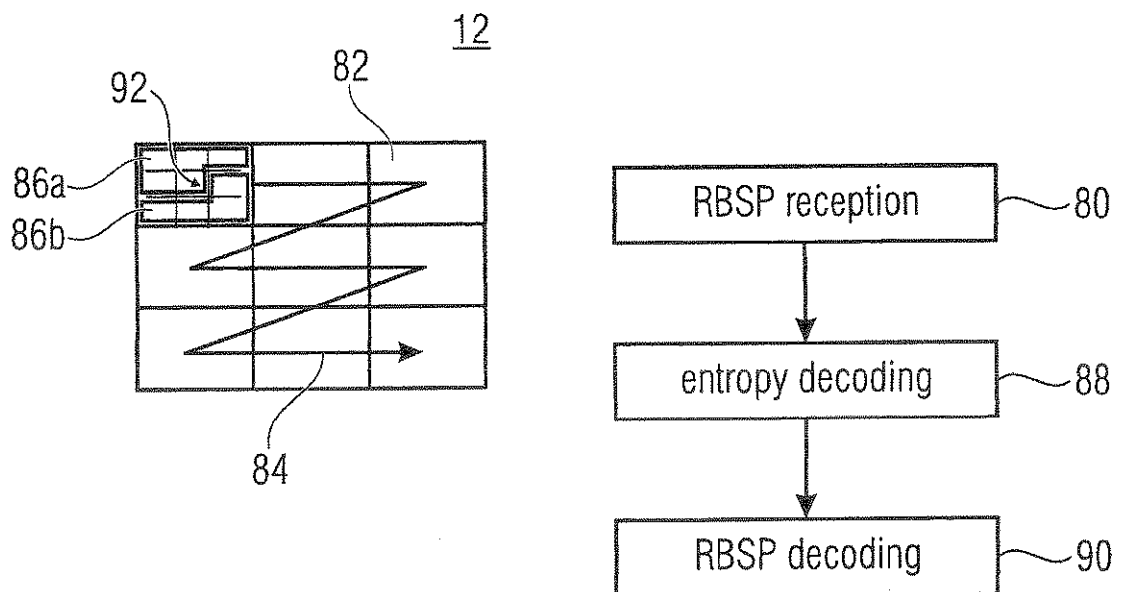
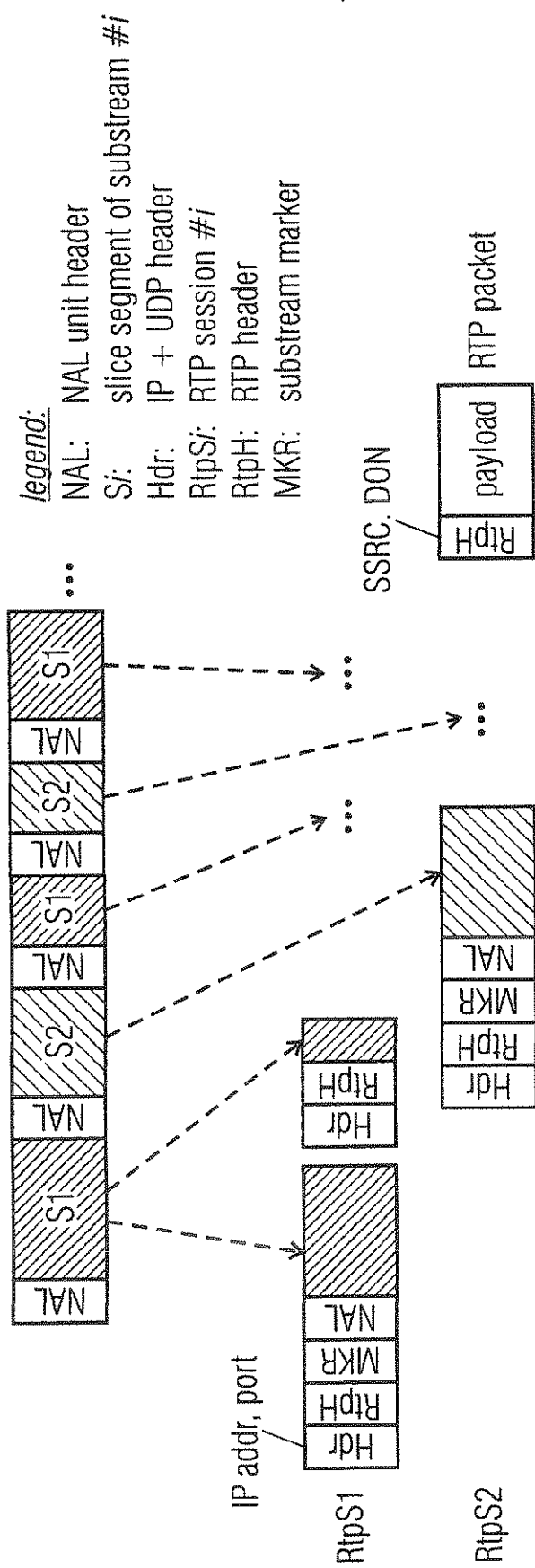


FIG 18



9154