

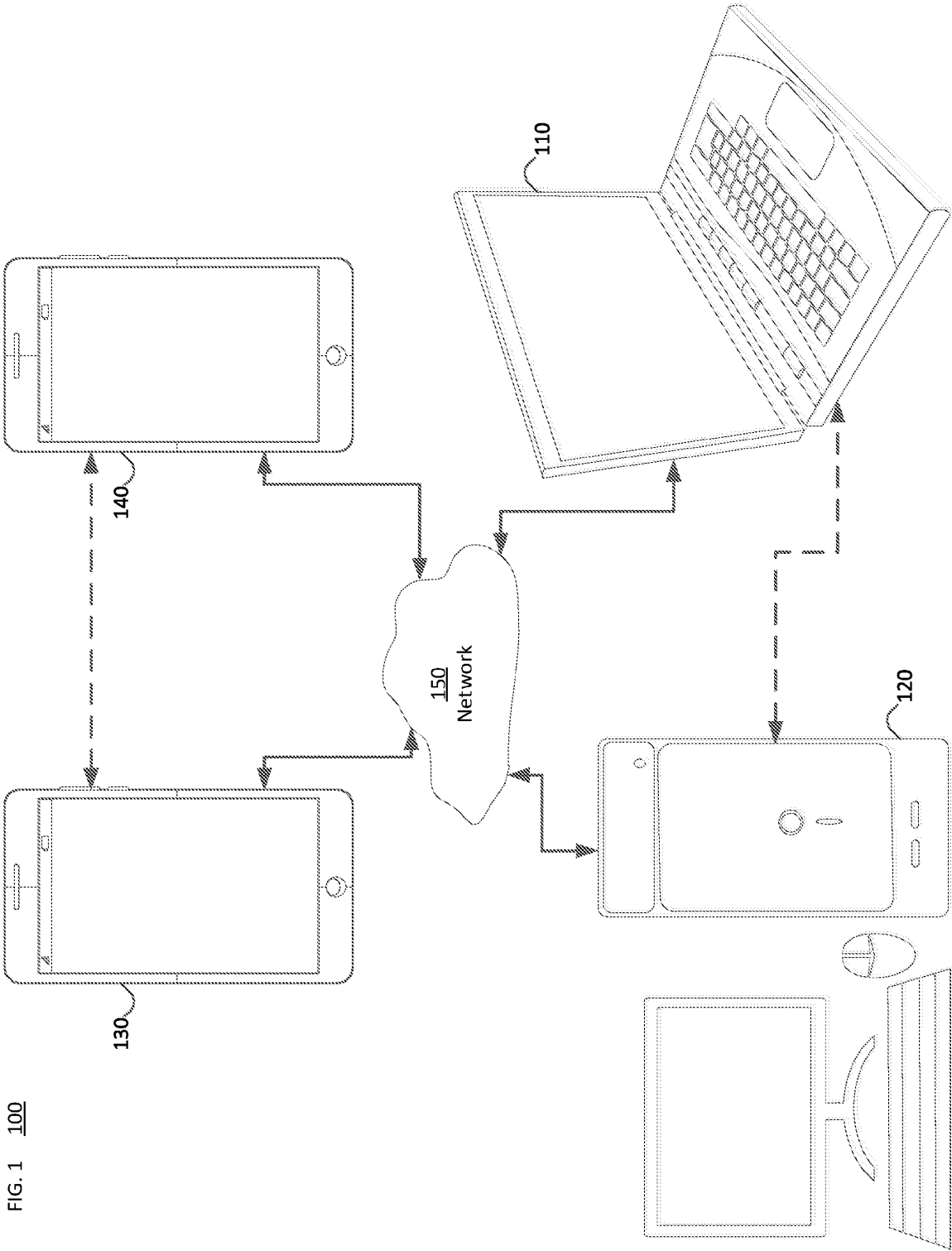


FIG. 1 100

FIG. 2 Streaming System 200

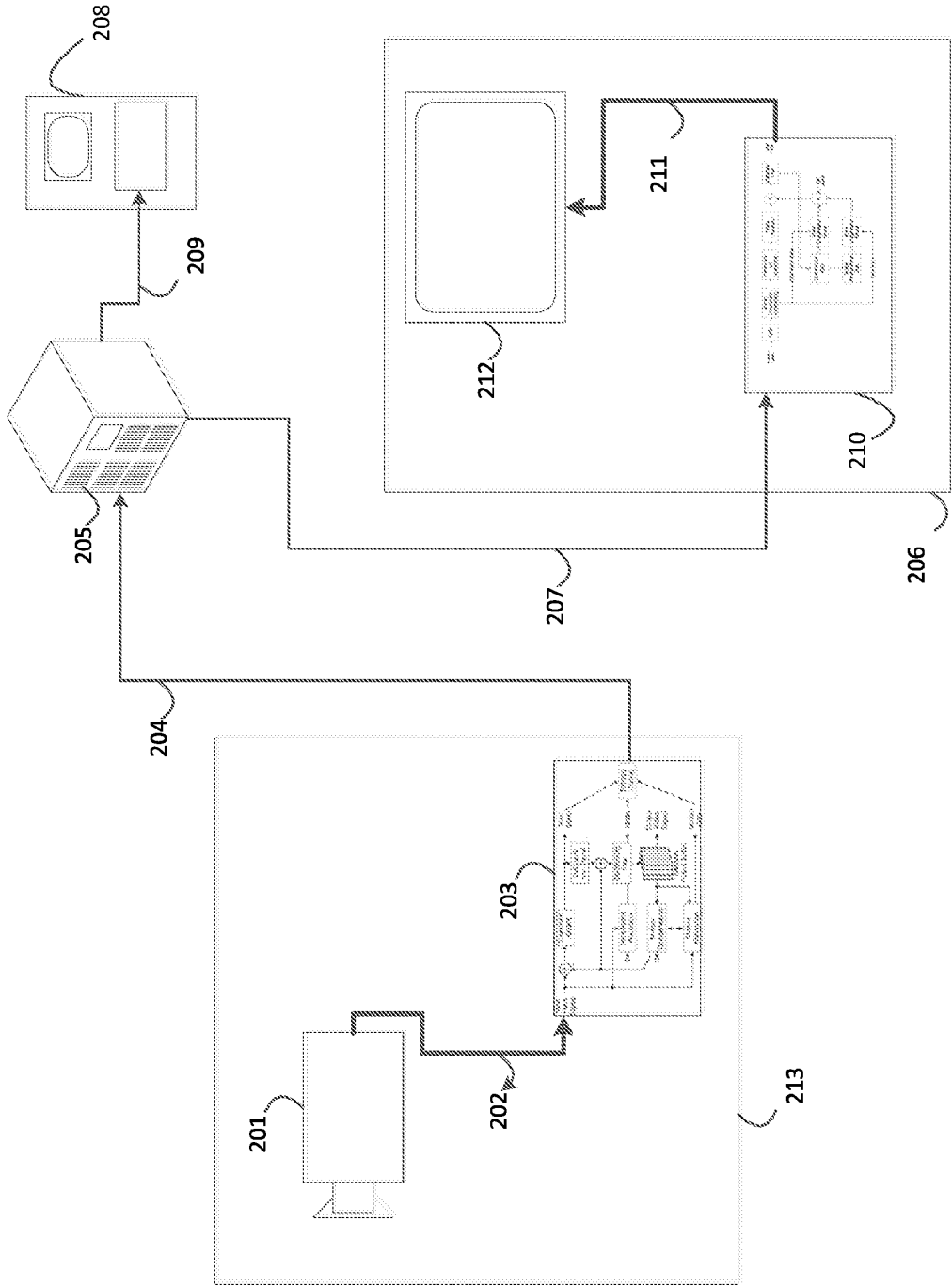


FIG. 3 Decoder 210

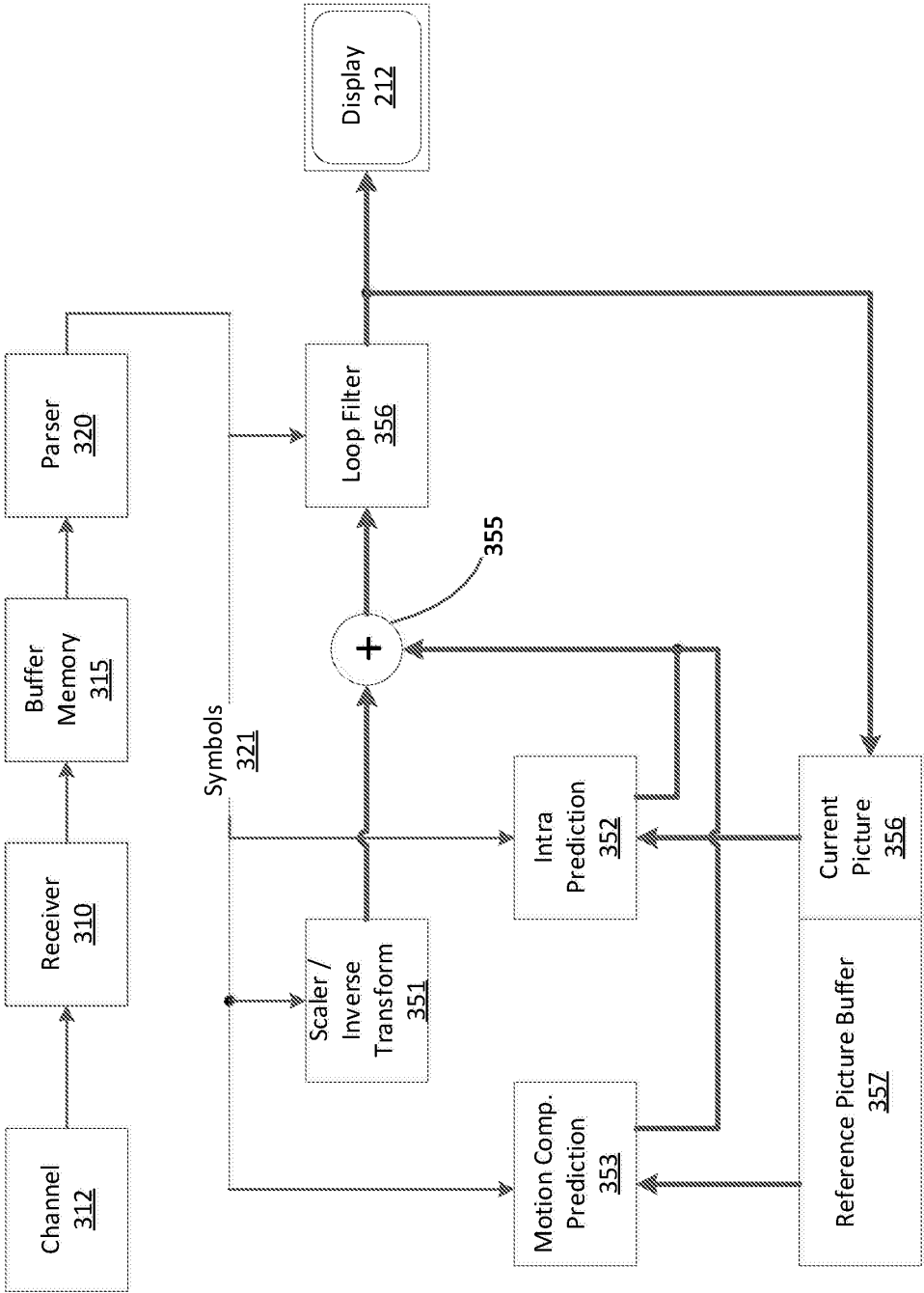
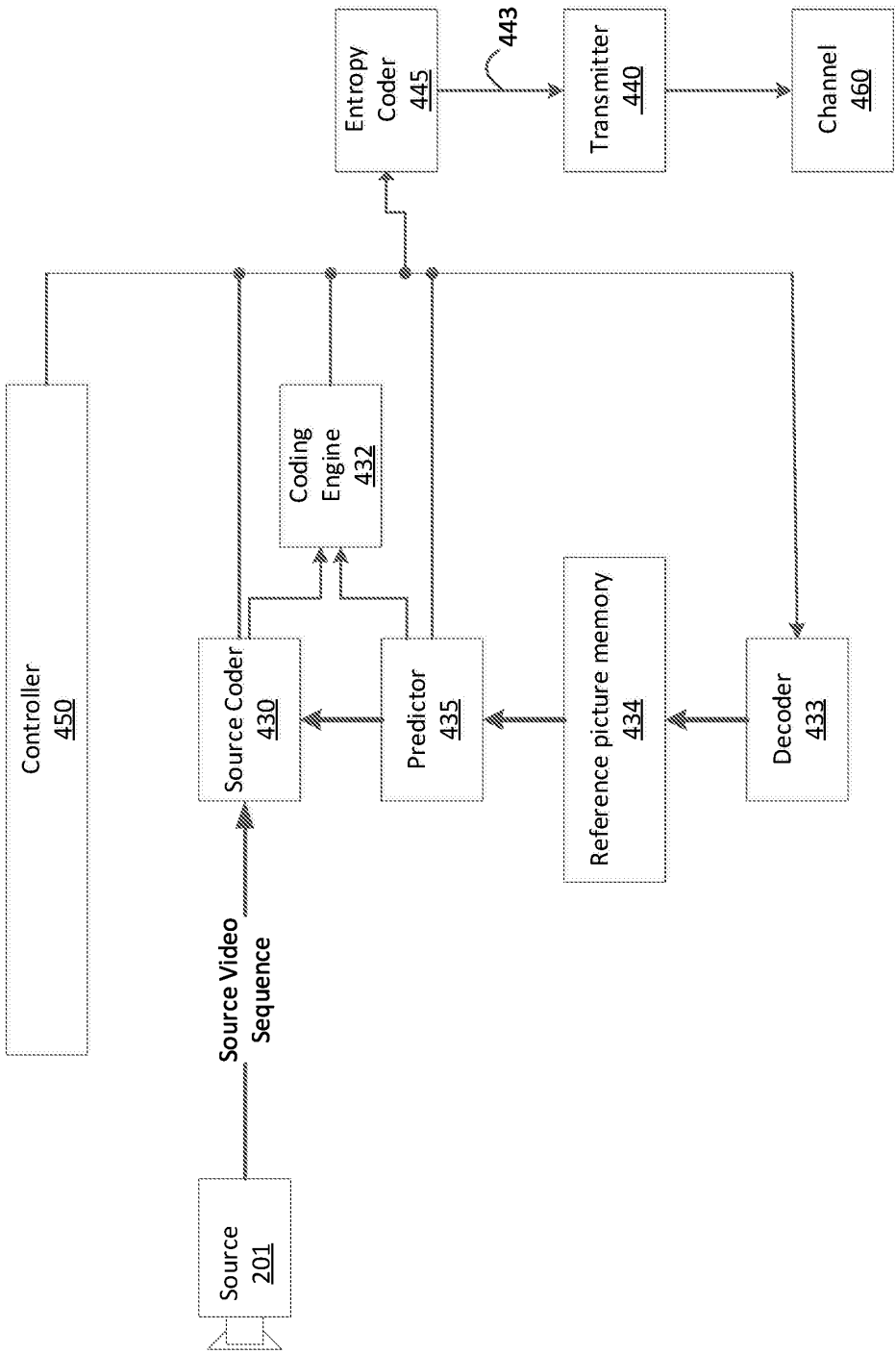


FIG. 4 Encoder 203



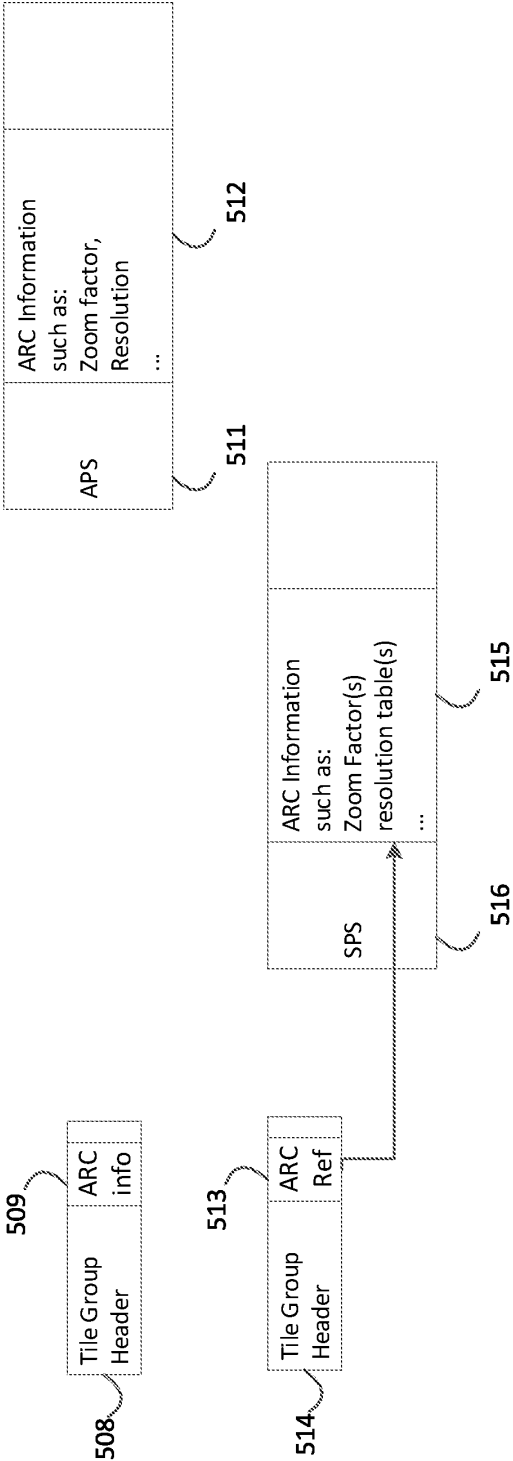
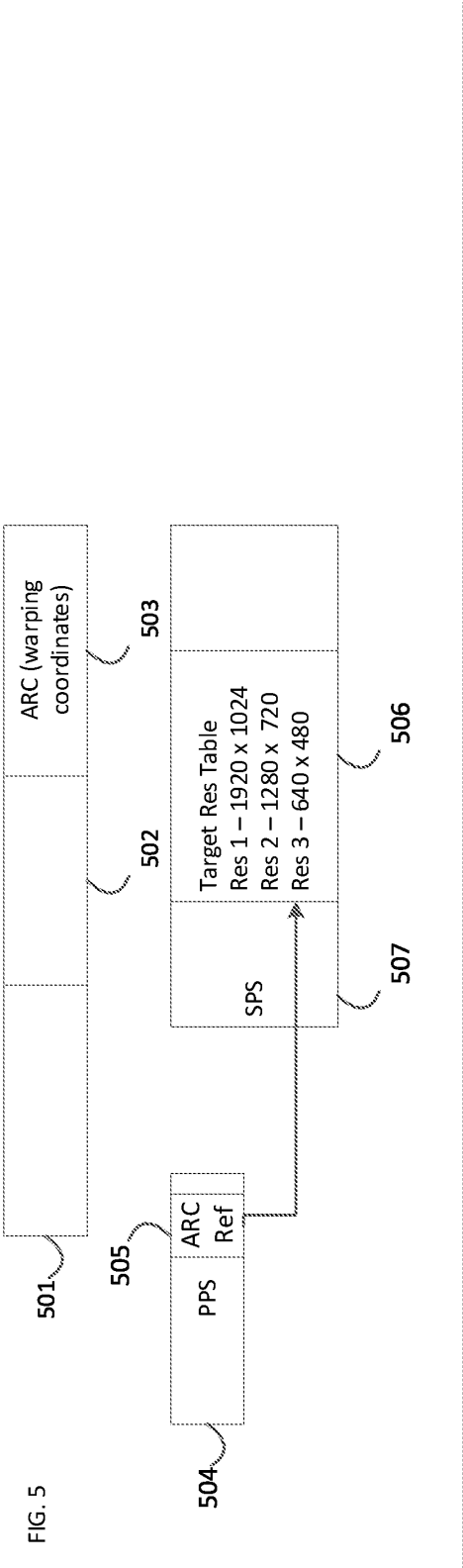
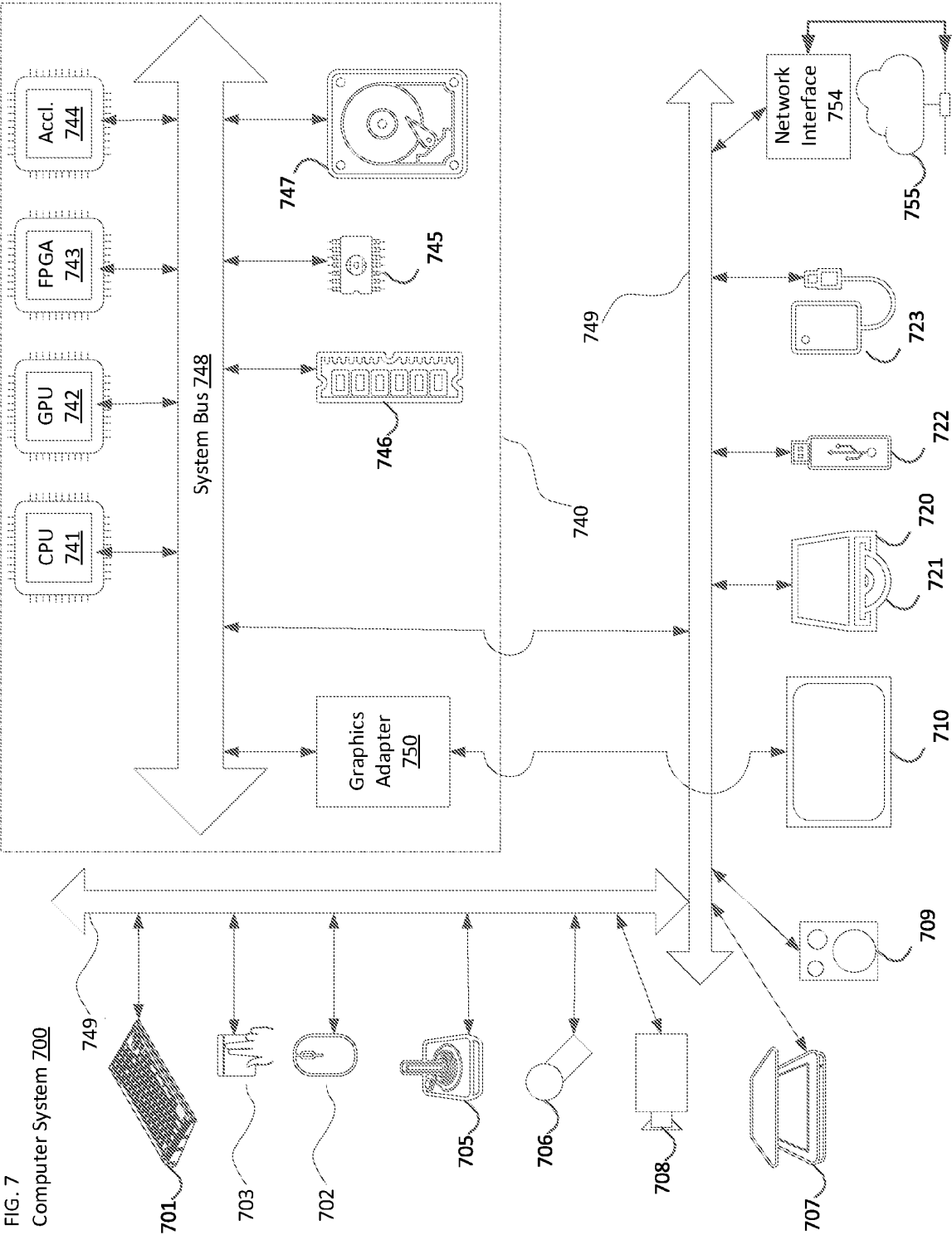
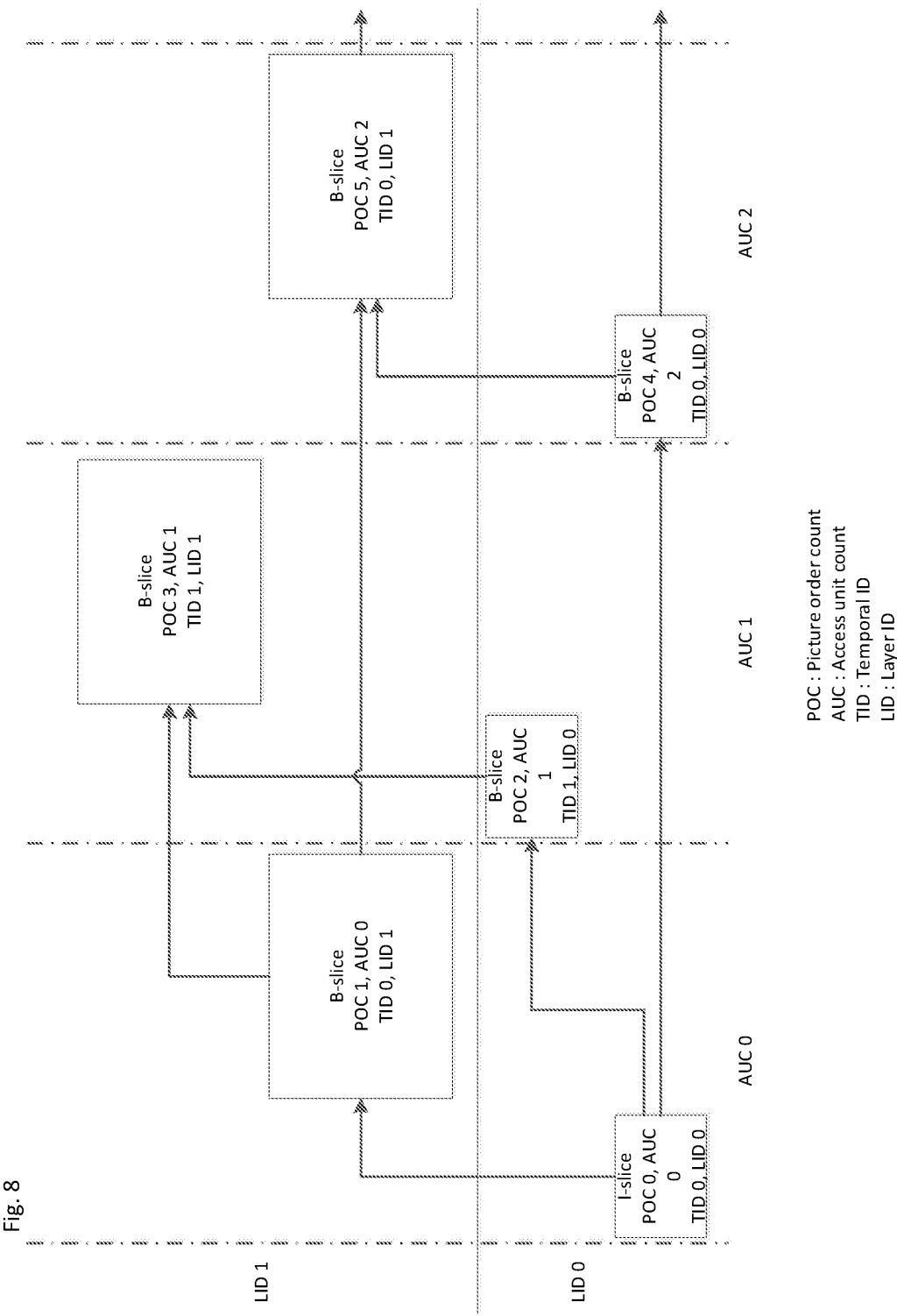


Fig. 6

601	tile_group_header() {		
	...		
603	if(adaptive_pic_resolution_change_flag) {		
602	dec_pic_size_idx		u(1)
	}		
	...		
	}		

610	seq_parameter_set_rbsp() {		Descriptor
	...		
611	adaptive_pic_resolution_change_flag		u(1)
612	if(adaptive_pic_resolution_change_flag) {		
	output_pic_width_in_luma_samples		ue(v)
613	output_pic_height_in_luma_samples		ue(v)
	reference_pic_size_present_flag		u(1)
614	if(reference_pic_size_present_flag)		
	{		
	reference_pic_width_in_luma_samples		ue(v)
615	reference_pic_height_in_luma_samples		ue(v)
	}		
	num_dec_pic_size_in_luma_samples_minus1		ue(v)
616	for(i = 0; i <= num_dec_pic_size_in_luma_samples_minus1; i++) {		
	dec_pic_width_in_luma_samples[i]		ue(v)
617	dec_pic_height_in_luma_samples[i]		ue(v)
	}		
	}		
	...		
	}		





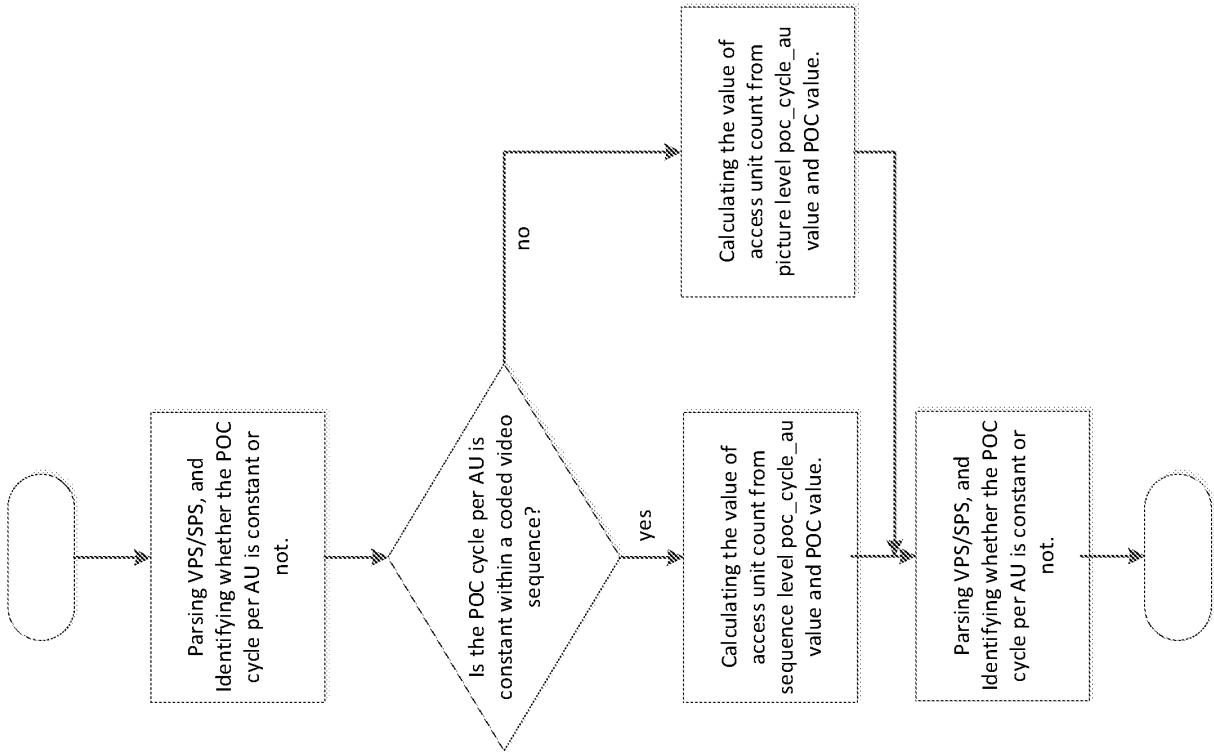


Fig. 10

FIG. 11

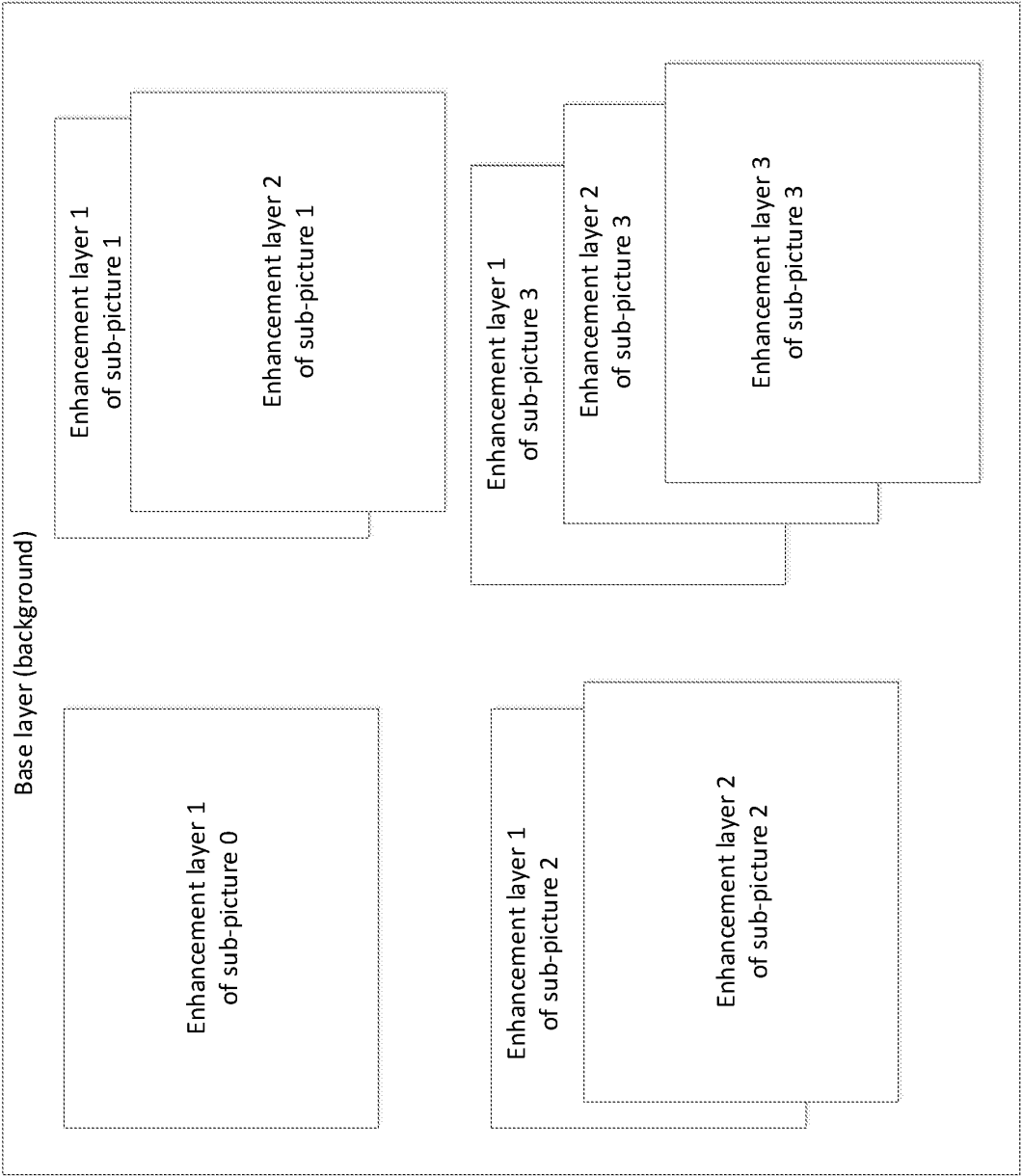


FIG. 12

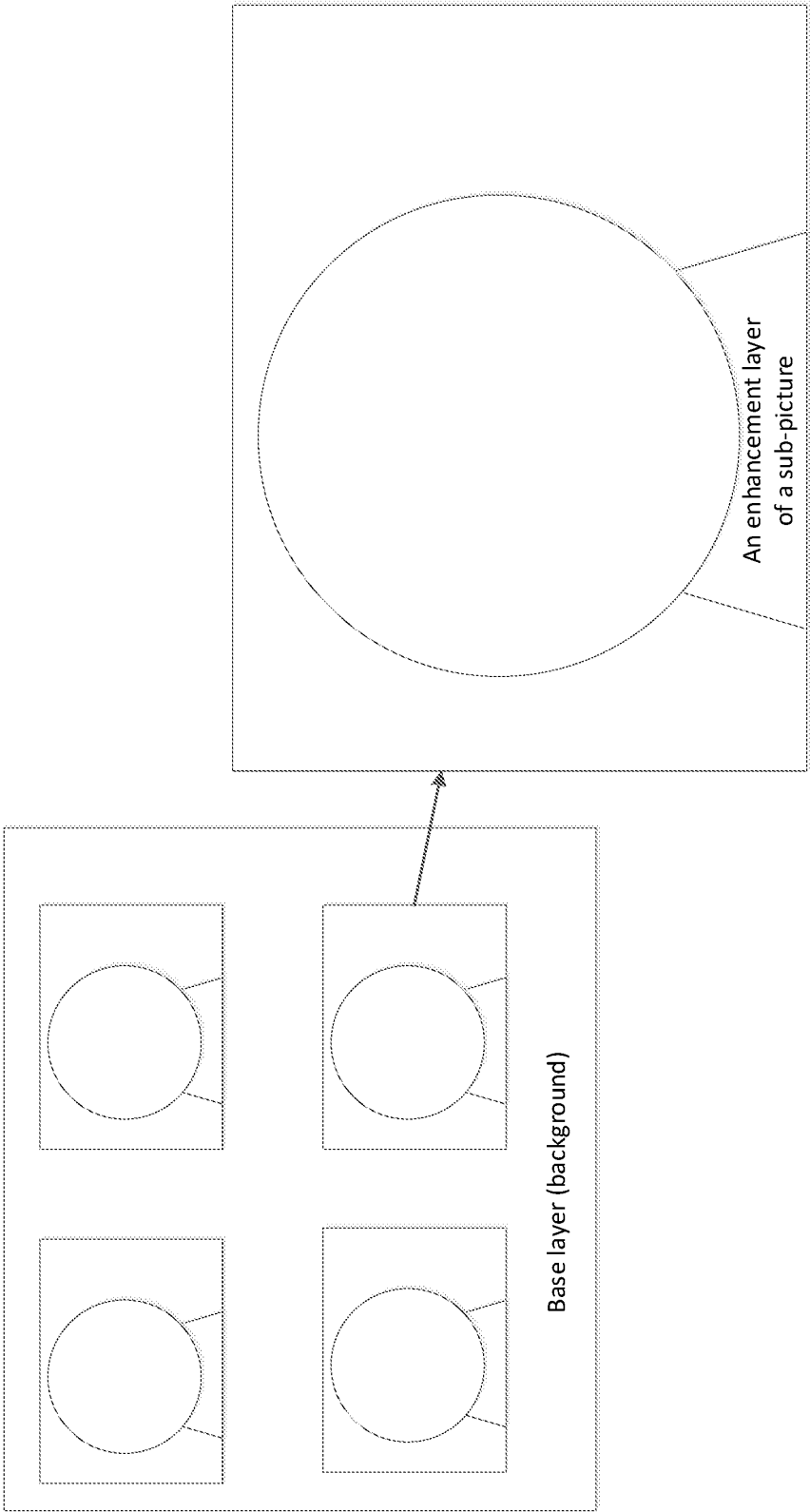


Fig. 13

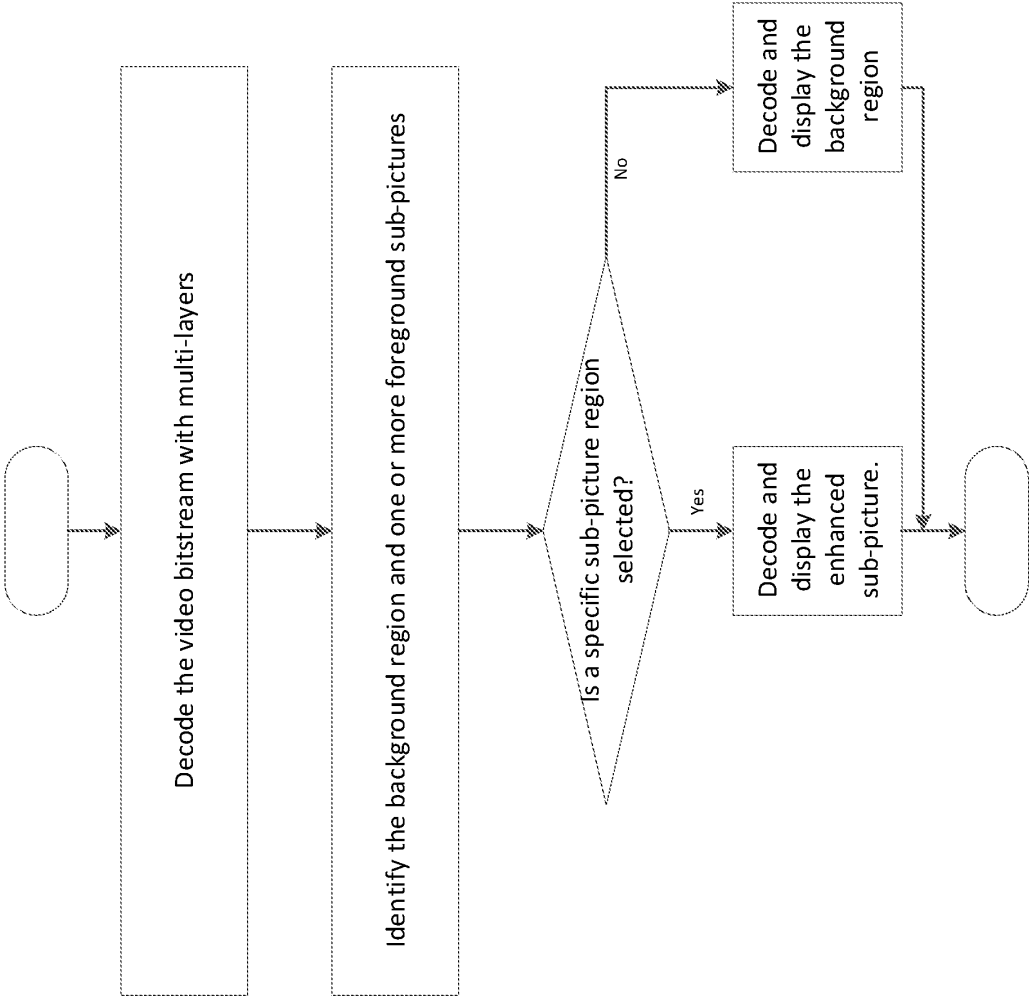


Fig. 14

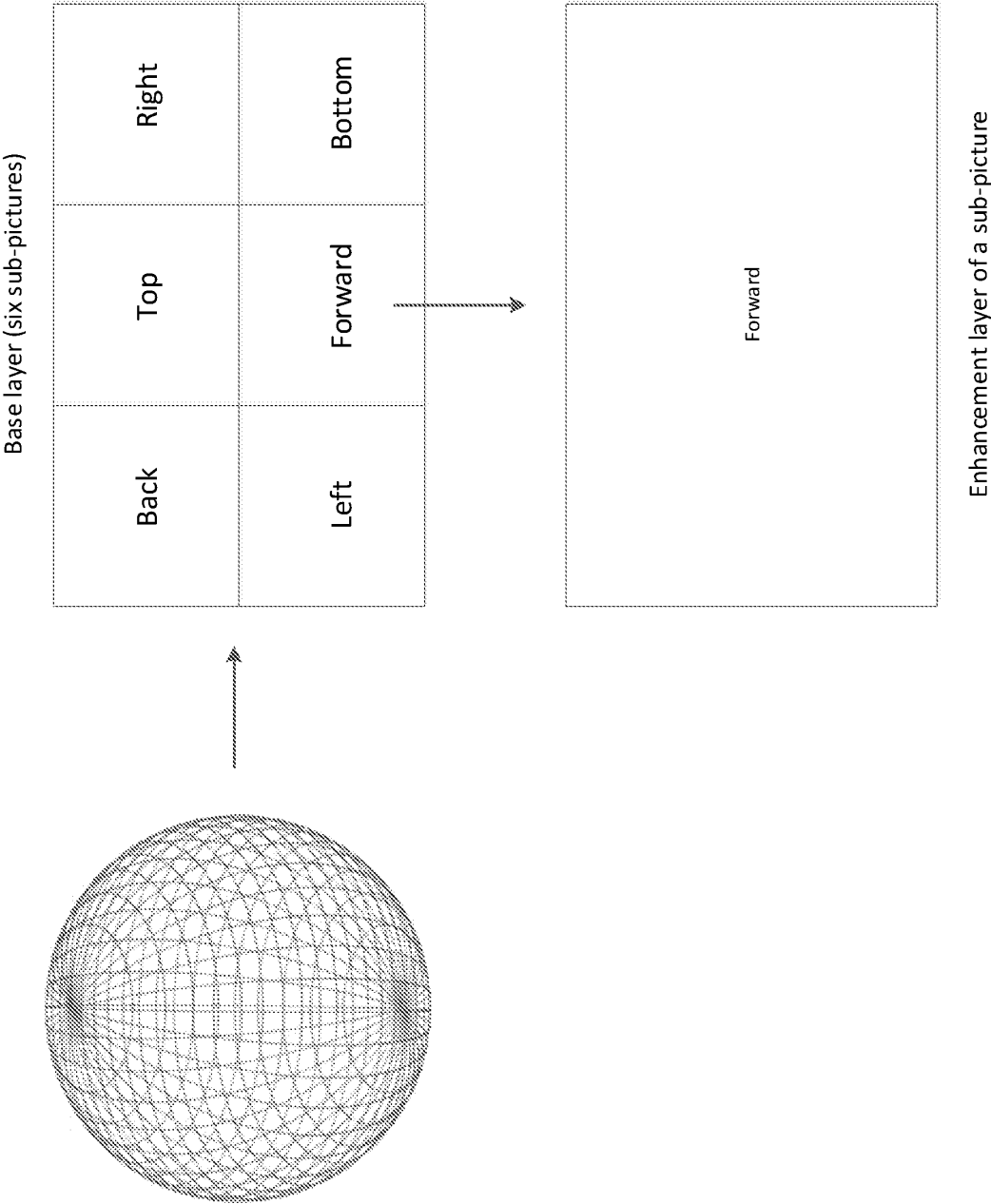
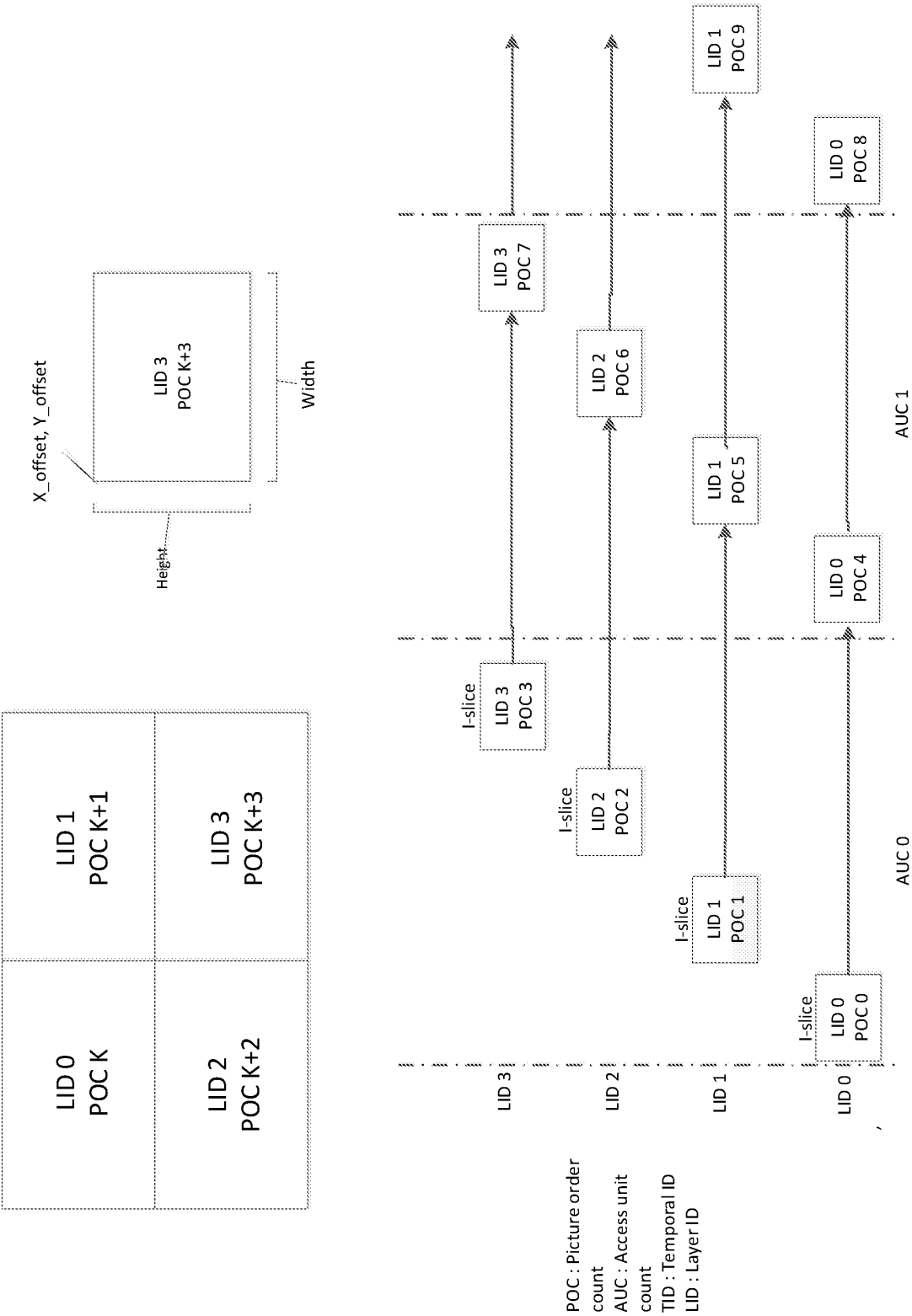


Fig. 15



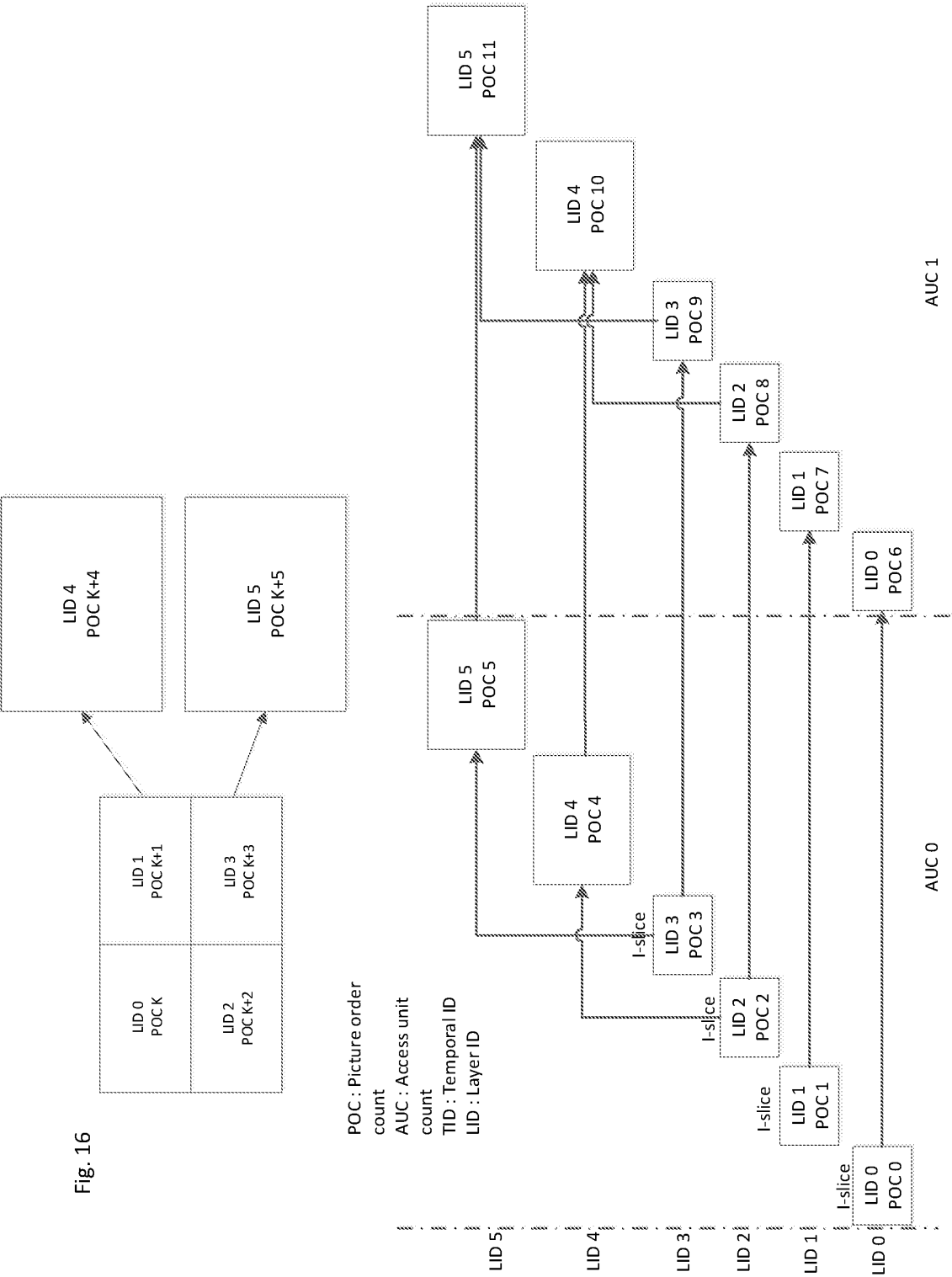


Fig. 16

Fig. 17

video_parameter_set_rbsp() {	Descriptor
aps_video_parameter_set_id	u(4)
vps_max_layers_minus1	u(8)
for(i = 0; i <= vps_max_layers_minus1; i++) {	
vps_included_layer_id[i]	u(7)
vps_reserved_zero_bit	u(1)
}	
...	
vps_sub_picture_dividing_flag	u(1)
if(vps_sub_picture_dividing_flag) {	
vps_full_pic_width_in_luma_samples	ue(v)
vps_full_pic_height_in_luma_samples	ue(v)
}	
...	
}	
*seq_parameter_set_rbsp() {	Descriptor
aps_decoding_parameter_set_id	u(4)
aps_video_parameter_set_id	u(4)
sps_max_sub_layers_minus1	u(3)
...	
pic_width_in_luma_samples	ue(v)
pic_height_in_luma_samples	ue(v)
if(vps_sub_picture_dividing_flag) {	
pic_offset_x	ue(v)
pic_offset_y	ue(v)
}	
...	
}	

Fig. 18

sub_region_partitioning_info(payloadSize) { ^o		Descriptor ⁺⁺
num_sub_region ^o		yes(v) ^o
num_layers ^o		yes(v) ^o
for(i = 0; i <= num_layers; i++) ^o		+ ^o
layer_id [i] ^o		yes(v) ^o
for(j = 1; j < num_layers; j++) ^o		+ ^o
for(i = 0; j < i; i++) ^o		+ ^o
direct_dependency_flag [i] [j] ^o		u(1) ^o
for(i = 0; i < num_sub_region; i++) { ^o		+ ^o
num_layers_for_region [i] ^o		yes(v) ^o
for(i = 0; i < num_layers_for_region [i]; i++) ^o		+ ^o
sub_region_layer_id [i] [i] ^o		yes(v) ^o
sub_region_offset_x [i] ^o		yes(v) ^o
sub_region_offset_y [i] ^o		yes(v) ^o
sub_region_width [i] ^o		yes(v) ^o
sub_region_height [i] ^o		yes(v) ^o
} ^o		+ ^o
... ^o		+ ^o
} ^o		+ ^o

Fig. 19

	Descriptor
video parameter set rsp() {	
...	
vps max layers minus1	u(6)
num output layer sets	ue(v)
num profile tier level	ue(v)
for(i = 0; i < num profile tier level; i++)	
profile tier level(vps max sub layers minus1)	
for(i = 0; i < num output layer sets; i++)	
for(j = 0; j < NumLayersInIdList[i]; j++) {	
output layer flag[i][j]	u(1)
profile tier level idx[i][j]	u(v)
}	
...	
}	

Fig. 20

video_parameter_set_rbsp() {	Descriptor
...	
vps_max_layers_minus1	u(6)
if(vps_max_layers_minus1 > 0) {	
num_output_layer_sets	ue(v)
num_profile_tier_level	ue(v)
}	
for(i = 0; i < num_profile_tier_level; i++)	
profile_tier_level(vps_max_sub_layers_minus1)	
for(i = 0; i < num_output_layer_sets; i++) {	
vps_output_layers_mode[i]	u(2)
vps_ptl_signal_flag[i]	u(1)
for(j = 0; j < NumLayersInIdList[i]; j++) {	
if(vps_output_layers_mode[i] == 2)	
output_layer_flag[i][j]	u(1)
if(vps_ptl_signal_flag[i])	
profile_tier_level_idx[i][j]	u(v)
}	
}	
...	
}	

Fig. 21

	Descriptor
video_parameter_set_rbsp() {	
...	
vps_max_layers_minus1	u(6)
if(vps_max_layers_minus1 > 0) {	
num_output_layer_sets	ue(v)
num_profile_tier_level	ue(v)
}	
max_subpics_minus1	u(8)
for(i = 0; i < max_subpics_minus1; i++) {	
sub_pic_id[i]	u(8)
}	
for(i = 0; i < num_profile_tier_level; i++)	
profile_tier_level(vps_max_sub_layers_minus1)	
for(i = 0; i < num_output_layer_sets; i++) {	
vps_output_layers_mode[i]	u(2)
vps_ptl_signal_flag[i]	u(1)
for(j = 0; j < NumLayersInIdList[i]; j++) {	
num_output_subpic_layer[i][j]	ue(v)
for(k = 0; k < num_output_subpic_layer[i][j]; k++)	
sub_pic_id_layer[i][j][k]	u(8)
if(vps_output_layers_mode[i] == 2)	
output_layer_flag[i][j]	u(1)
if(vps_ptl_signal_flag[i])	
profile_tier_level_idx[i][j]	u(v)
}	
}	
...	
}	

Fig. 22

	Descriptor
pic parameter set rbsp() {	
pps pic parameter set id	ue(v)
pps seq parameter set id	u(4)
mixed nalu types in pic flag	u(1)
pic width in luma samples	ue(v)
pic height in luma samples	ue(v)
pps conformance window flag	u(1)
if(pps conformance window flag) {	
pps conf win left offset	ue(v)
pps conf win right offset	ue(v)
pps conf win top offset	ue(v)
pps conf win bottom offset	ue(v)
}	
scaling window explicit signalling flag	u(1)
if(scaling window explicit signalling flag) {	
scaling win left offset	ue(v)
scaling win right offset	ue(v)
scaling win top offset	ue(v)
scaling win bottom offset	ue(v)
}	
output flag present flag	u(1)
subpic id mapping in pps flag	u(1)
if(subpic id mapping in pps flag) {	
pps num subpics minus 1	ue(v)
pps subpic id len minus 1	ue(v)
for(i = 0; i <= pps num subpic minus 1; i++)	
pps subpic id[i]	u(v)
}	
...	
}	

Fig. 23

	Descriptor
seq_parameter_set_rbsp() {	
...	
subpic_info_present_flag	u(2)
if(subpic_info_present_flag) {	u(1)
sps_num_subpics_minus1	
sps_independent_subpics_flag	ue(v)
for(i = 0; sps_num_subpics_minus1 > 0 && i <= sps_num_subpics_minus1; i++) {	u(1)
if(i > 0 && pic_width_max_in_luma_samples > CtbSizeY)	
subpic_ctu_top_left_x[i]	
if(i > 0 && pic_height_max_in_luma_samples > CtbSizeY) {	u(v)
subpic_ctu_top_left_y[i]	
if(i < sps_num_subpics_minus1 && pic_width_max_in_luma_samples > CtbSizeY)	u(v)
subpic_width_minus1[i]	
if(i < sps_num_subpics_minus1 && pic_height_max_in_luma_samples > CtbSizeY)	u(v)
subpic_height_minus1[i]	
if(!sps_independent_subpics_flag) {	
subpic_treated_as_pic_flag[i]	u(1)
loop_filter_across_subpic_enabled_flag[i]	u(1)
}	
}	
}	
}	ue(v)
sps_subpic_id_len_minus1	u(1)
subpic_id_mapping_explicitly_signalled_flag	
if(subpic_id_mapping_explicitly_signalled_flag) {	
subpic_id_mapping_in_sps_flag	u(1)
if(subpic_id_mapping_in_sps_flag)	
for(i = 0; i <= sps_num_subpics_minus1; i++)	
sps_subpic_id[i]	u(v)
}	
}	
...	ue(v)
}	

Fig. 24

pic parameter set rbsp() {	Descriptor
...	
pps_num_subpics_minus1	ue(v)
subpic_id_mapping_in_pps_flag	u(1)
if(subpic_id_mapping_in_pps_flag) {	
pps_subpic_id_len_minus1	ue(v)
for(i = 0; i <= pps_num_subpic_minus1; i++)	
pps_subpic_id[i]	u(v)
}	
no_pic_partition_flag	u(1)
if(!no_pic_partition_flag) {	
pps_log2_ctu_size_minus5	u(2)
num_exp_tile_columns_minus1	ue(v)
num_exp_tile_rows_minus1	ue(v)
for(i = 0; i <= num_exp_tile_columns_minus1; i++)	
tile_column_width_minus1[i]	ue(v)
for(i = 0; i <= num_exp_tile_rows_minus1; i++)	
tile_row_height_minus1[i]	ue(v)
if(NumTilesInPic > 1)	
rect_slice_flag	u(1)
if(rect_slice_flag)	
single_slice_per_subpic_flag	u(1)
if(rect_slice_flag && !single_slice_per_subpic_flag) {	
num_slices_in_pic_minus1	ue(v)
if(num_slices_in_pic_minus1 > 0)	
tile_idx_delta_present_flag	u(1)
for(i = 0; i < num_slices_in_pic_minus1; i++) {	
if(NumTileColumns > 1)	
slice_width_in_tiles_minus1[i]	ue(v)
if(NumTileRows > 1 && (tile_idx_delta_present_flag	
SliceTopLeftTileIdx[i] % NumTileColumns == 0))	
slice_height_in_tiles_minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 &&	
slice_height_in_tiles_minus1[i] == 0 &&	
RowHeight[SliceTopLeftTileIdx[i] / NumTileColumns > 1) {	
num_exp_slices_in_tile[i]	ue(v)
for(j = 0; j < num_exp_slices_in_tile[i]; j++)	
exp_slice_height_in_ctus_minus1[i][j]	ue(v)
i += NumSlicesInTile[i] - 1	
}	
}	
if(tile_idx_delta_present_flag && i < num_slices_in_pic_minus1)	
tile_idx_delta[i]	se(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
loop_filter_across_slices_enabled_flag	u(1)
}	
...	
}	

Fig. 25

pic_parameter_set_rbsp() {	Descriptor
...	
pps num subpics minus1	ue(v)
subpic id mapping in pps flag	u(1)
if(subpic id mapping in pps flag) {	
pps subpic id len minus1	ue(v)
for(i = 0; i <= pps num subpic minus1; i++)	
pps subpic id[i]	u(v)
}	
no pic partition flag	u(1)
if(!no pic partition flag) {	
pps log2 ctu size minus5	u(2)
num exp tile columns minus1	ue(v)
num exp tile rows minus1	ue(v)
for(i = 0; i <= num exp tile columns minus1; i++)	
tile column width minus1[i]	ue(v)
for(i = 0; i <= num exp tile rows minus1; i++)	
tile row height minus1[i]	ue(v)
if(NumTilesInPic > 1)	
rect slice flag	u(1)
if(rect slice flag)	
single slice per subpic flag	u(1)
if(rect slice flag && !single slice per subpic flag) {	
num_slices_in_pic_minus_num_sub_pics	ue(v)
if(NumSlicesInPic > 1)	
tile idx delta present flag	u(1)
for(i = 0; i < NumSlicesInPic-1; i++) {	
if(NumTileColumns > 1)	
slice width in tiles minus1[i]	ue(v)
if(NumTileRows > 1 && (tile_idx_delta_present_flag SliceTopLeftTileIdx[i] % NumTileColumns == 0))	
slice height in tiles minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0 && RowHeight[SliceTopLeftTileIdx[i] / NumTileColumns] > 1) {	
num_exp_slices_in_tile[i]	ue(v)
for(j = 0; j < num_exp_slices_in_tile[i]; j++)	
exp_slice_height_in_ctus_minus1[i][j]	ue(v)
i += NumSlicesInTile[i] - 1	
}	
if(tile_idx_delta_present_flag && i < NumSlicesInPic - 1)	
tile_idx_delta[i]	se(v)
}	
}	
loop_filter_across_tiles_enabled_flag	u(1)
loop_filter_across_slices_enabled_flag	u(1)
}	
...	
}	

Fig. 26

pic_parameter_set_rbsp() {	Descriptor
...	
pps num subpics minus1	ue(v)
subpic id mapping in pps flag	u(1)
if(subpic id mapping in pps flag) {	
pps subpic id len minus1	ue(v)
for(i = 0; i <= pps num subpic minus1; i++)	
pps subpic id[i]	u(v)
}	
no pic partition flag	u(1)
if(!no pic partition flag) {	
pps log2 ctu size minus5	u(2)
num exp tile columns minus1	ue(v)
num exp tile rows minus1	ue(v)
for(i = 0; i <= num exp tile columns minus1; i++)	
tile column width minus1[i]	ue(v)
for(i = 0; i <= num exp tile rows minus1; i++)	
tile row height minus1[i]	ue(v)
if(NumTilesInPic > 1)	
rect slice flag	u(1)
if(rect slice flag)	
single slice per subpic flag	u(1)
if(rect slice flag && !single slice per subpic flag) {	
num slices in pic minus num sub pics minus1	ue(v)
if(NumSlicesInPic > 1)	
tile idx delta present flag	u(1)
for(i = 0; i < NumSlicesInPic - 1; i++) {	
if(NumTileColumns > 1)	
slice width in tiles minus1[i]	ue(v)
if(NumTileRows > 1 && (tile_idx_delta_present_flag SliceTopLeftTileIdx[i] % NumTileColumns == 0))	
slice height in tiles minus1[i]	ue(v)
if(slice_width_in_tiles_minus1[i] == 0 && slice_height_in_tiles_minus1[i] == 0 && RowHeight[SliceTopLeftTileIdx[i] / NumTileColumns] > 1) {	
num exp slices in tile[i]	ue(v)
for(j = 0; j < num exp slices in tile[i]; j++)	
exp slice height in ctus minus1[i][j]	ue(v)
i += NumSlicesInTile[i] - 1	
}	
if(tile_idx_delta_present_flag && i < NumSlicesInPic - 1)	
tile idx delta[i]	se(v)
}	
}	
loop filter across tiles enabled flag	u(1)
loop filter across slices enabled flag	u(1)
}	
...	
}	