

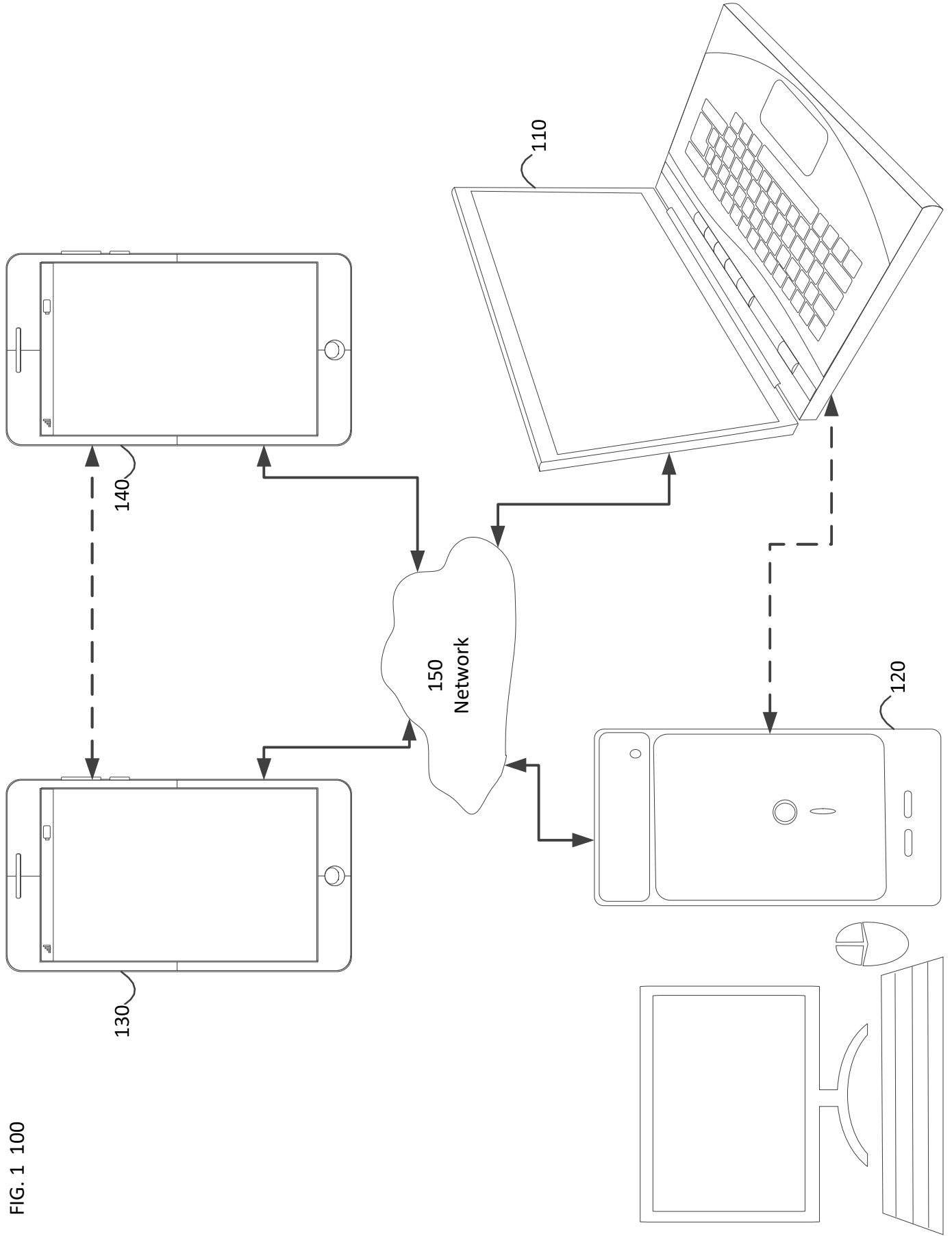


FIG. 1 100

FIG. 2

200

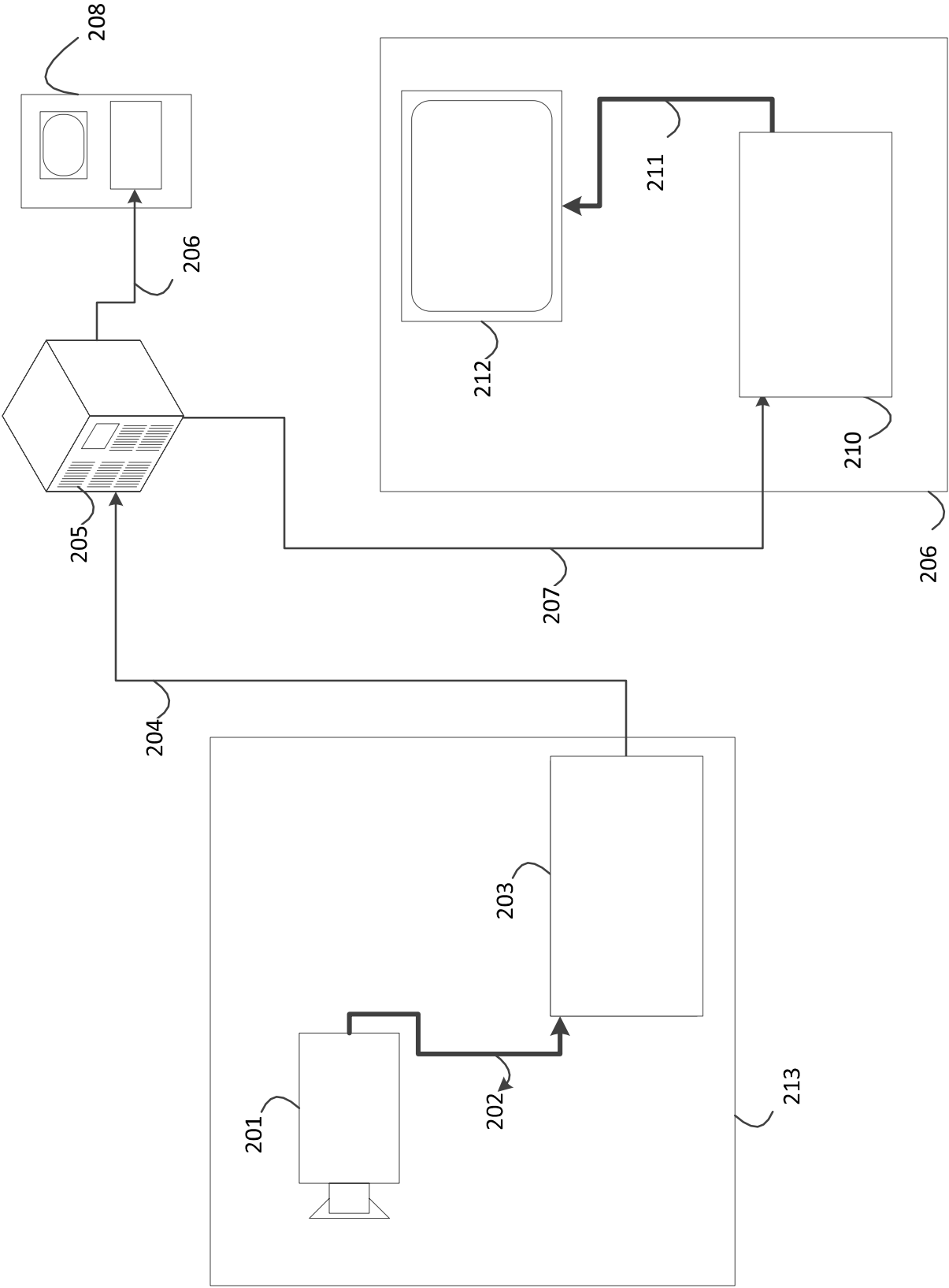


FIG. 3 Decoder 210

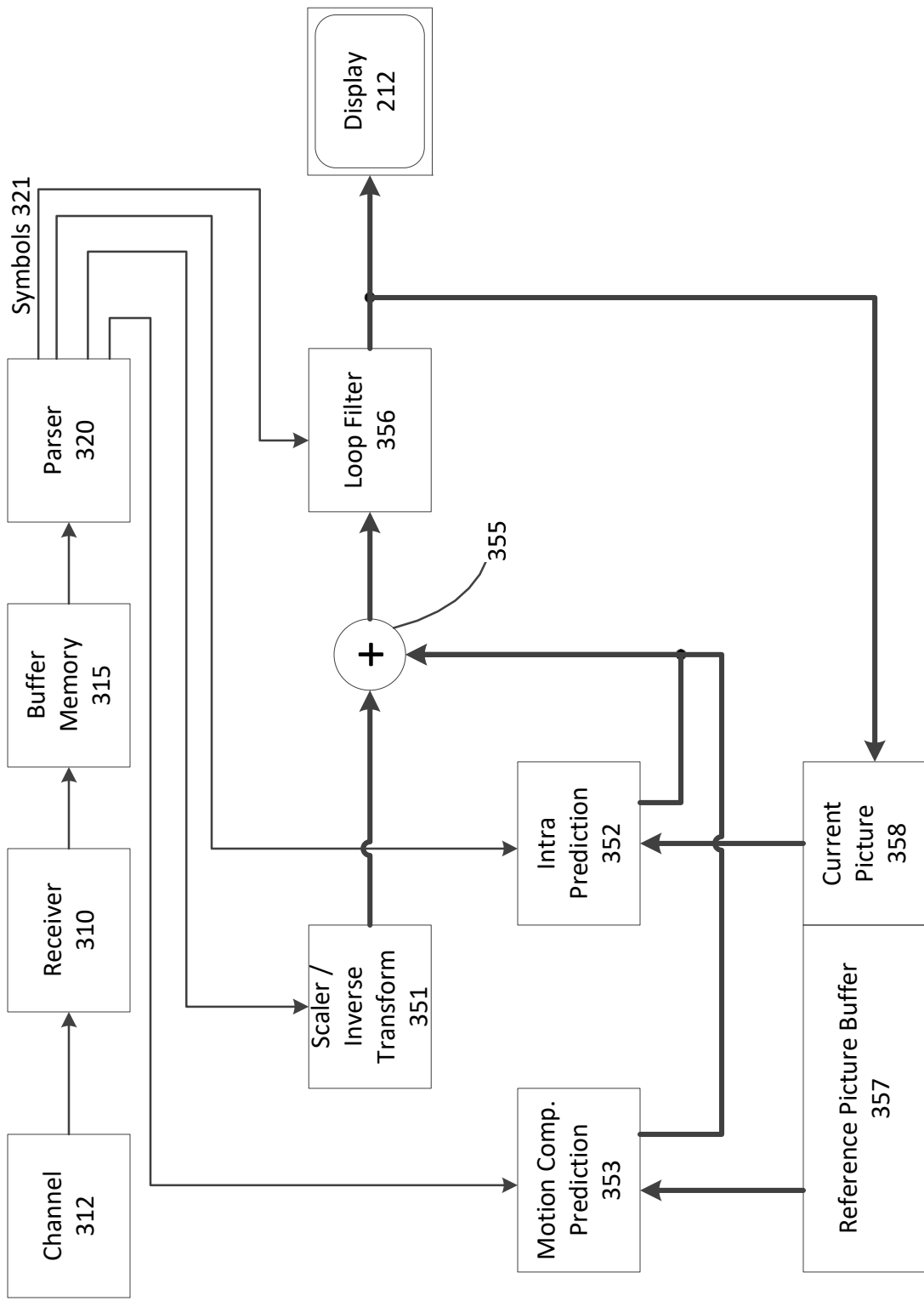
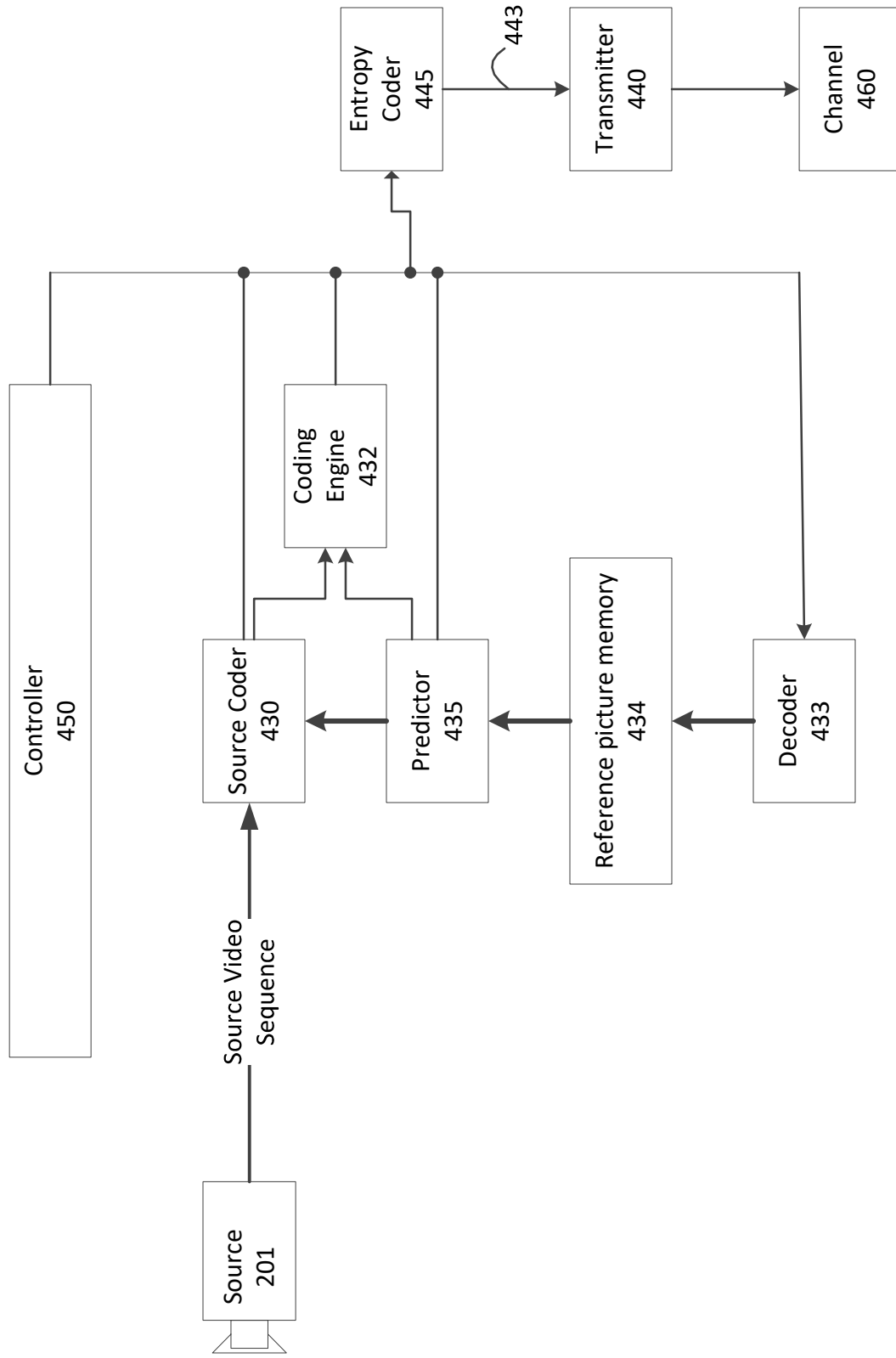


FIG. 4 Encoder 203

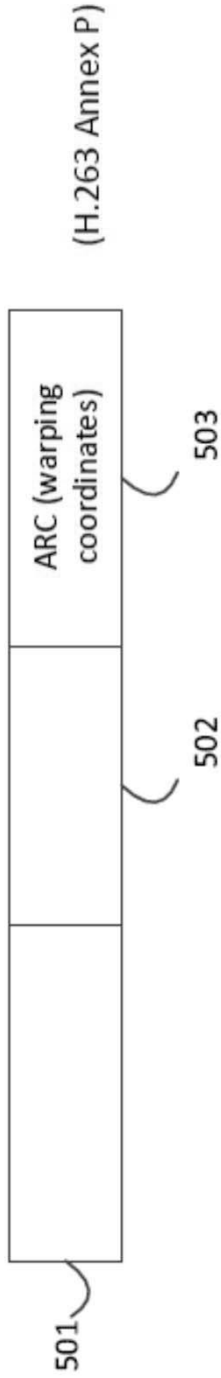


FIG. 5A

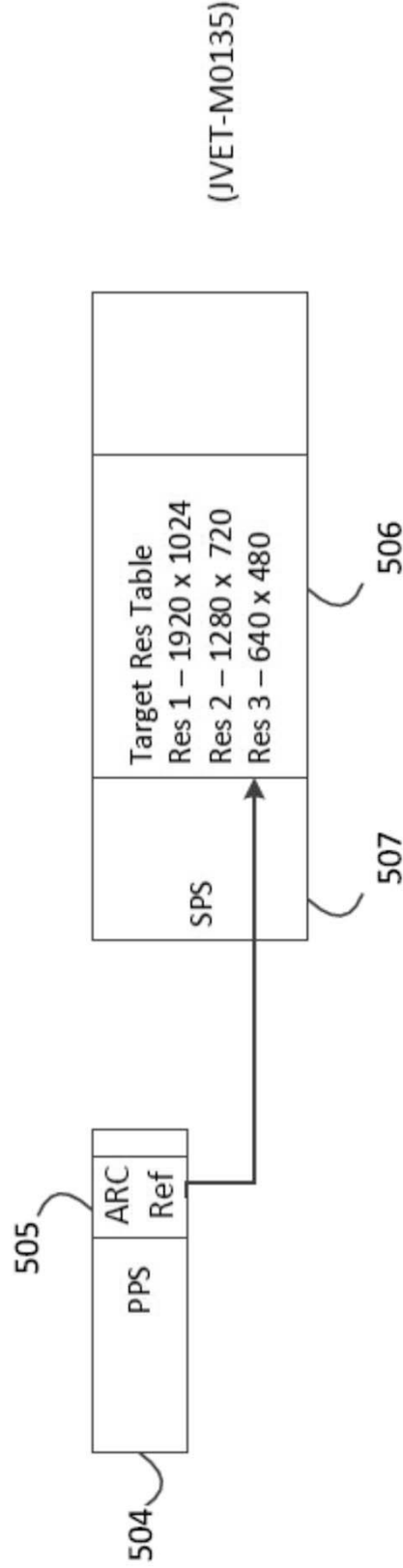


FIG. 5B



FIG. 5C



FIG. 5D

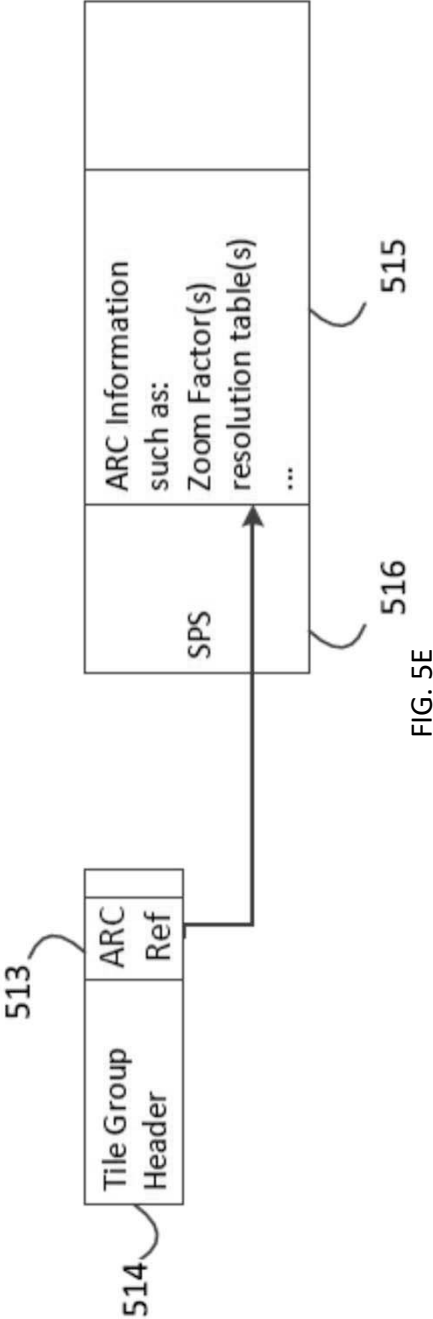


FIG. 5E

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

FIG. 6A

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

FIG. 6B

Fig. 7

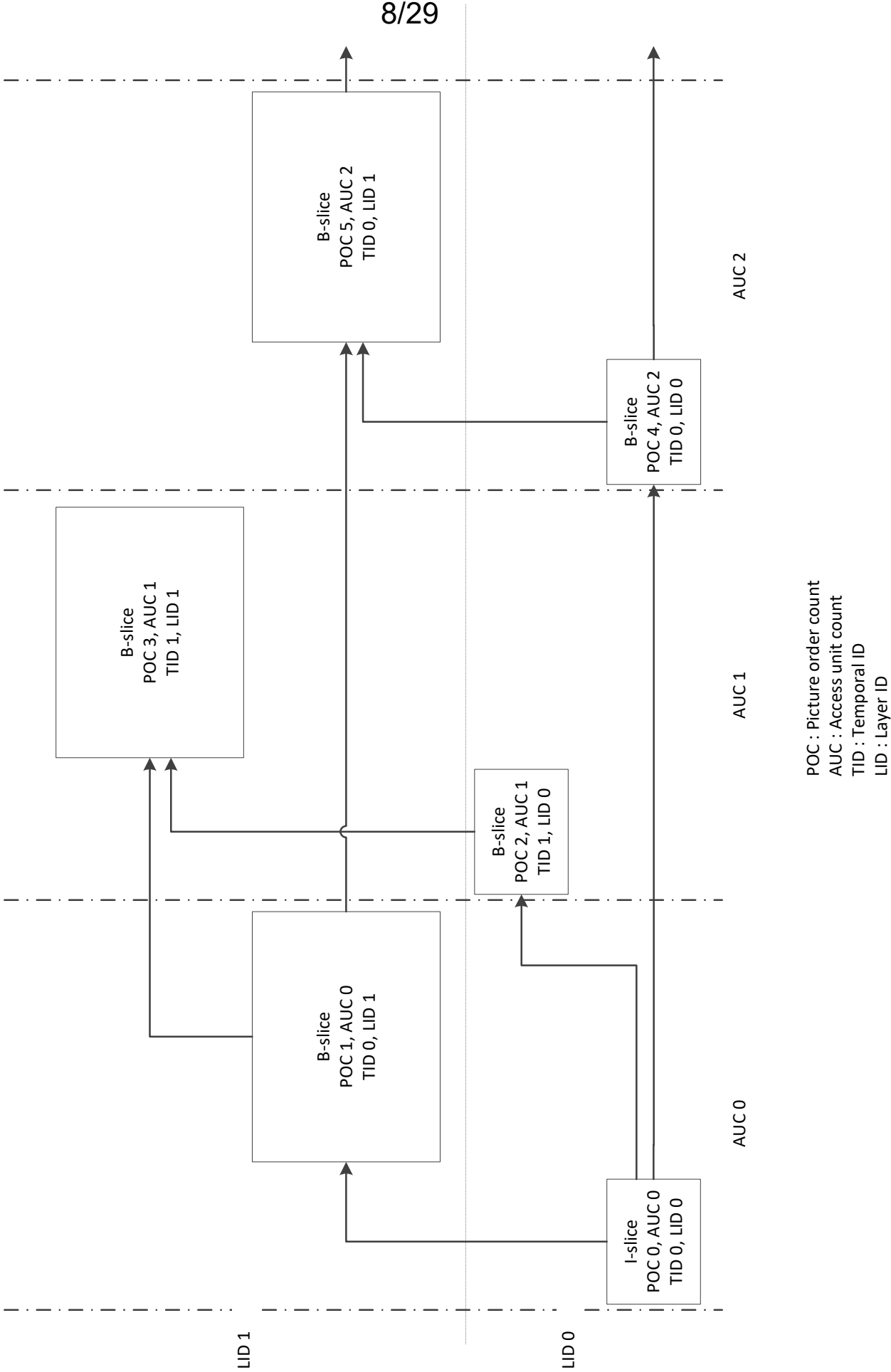


Fig. 8

	Descriptor
video_parameter_set_rbsp() {	
vps_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_constraint_info_present_flag	u(1)
vps_constant_poc_cycle_per_au	u(1)
if(vps_constant_poc_cycle_per_au)	
vps_poc_cycle_au	u(8)
...	
}	

	Descriptor
slice_header() {	
slice_pic_parameter_set_id	ue(v)
if(!rect_slice_flag NumBricksInPic > 1)	
slice_address	u(v)
if(!rect_slice_flag && !single_brick_per_slice_flag)	
num_bricks_in_slice_minus1	ue(v)
slice_type	ue(v)
if(!NalUnitType == GRA_NUT)	
recovery_poc_cnt	se(v)
slice_pic_order_cnt_lsb	u(v)
...	
if(!vps_constant_poc_cycle_per_au)	
slice_poc_cycle_au	u(8)
}	

Fig. 9

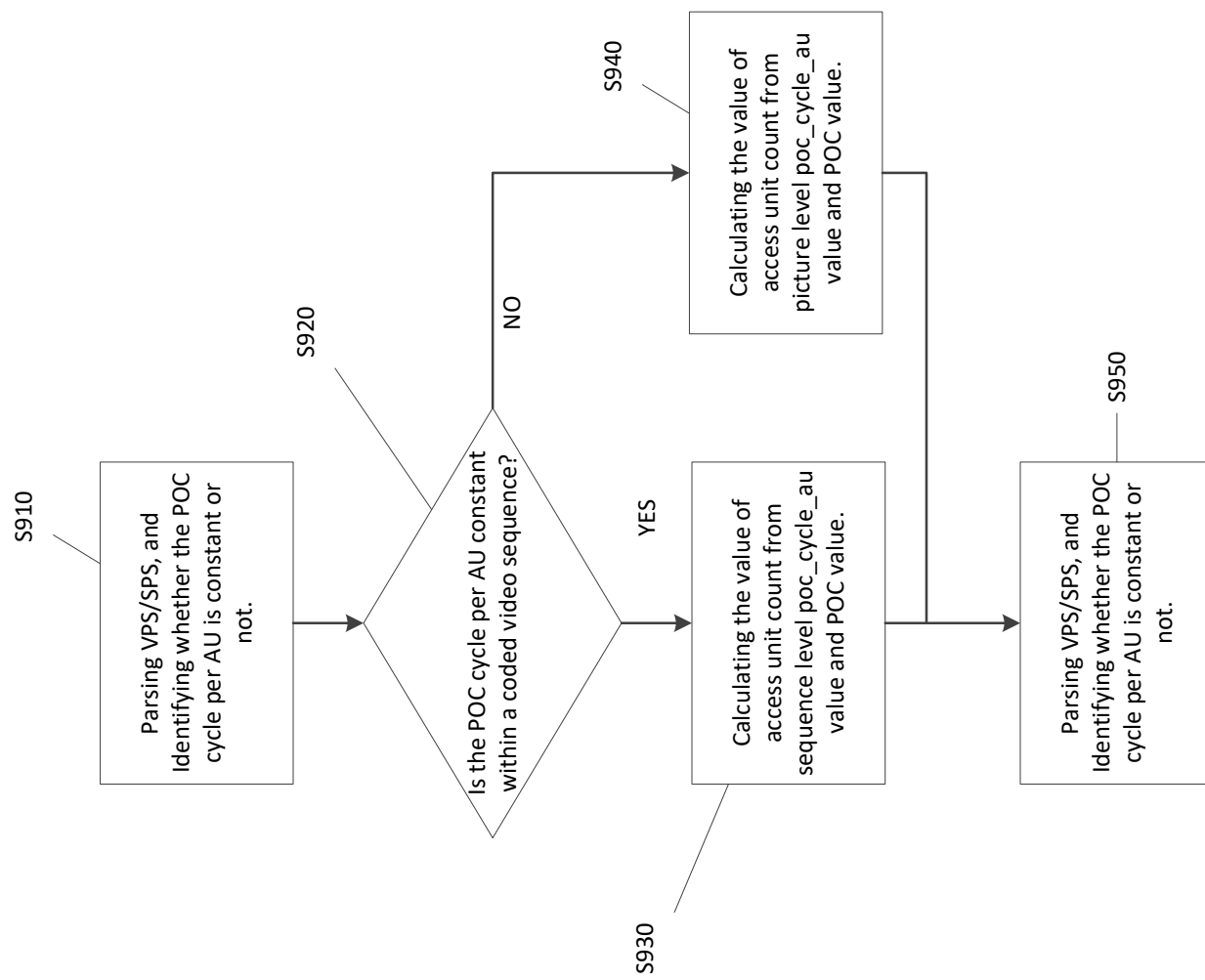


FIG. 10

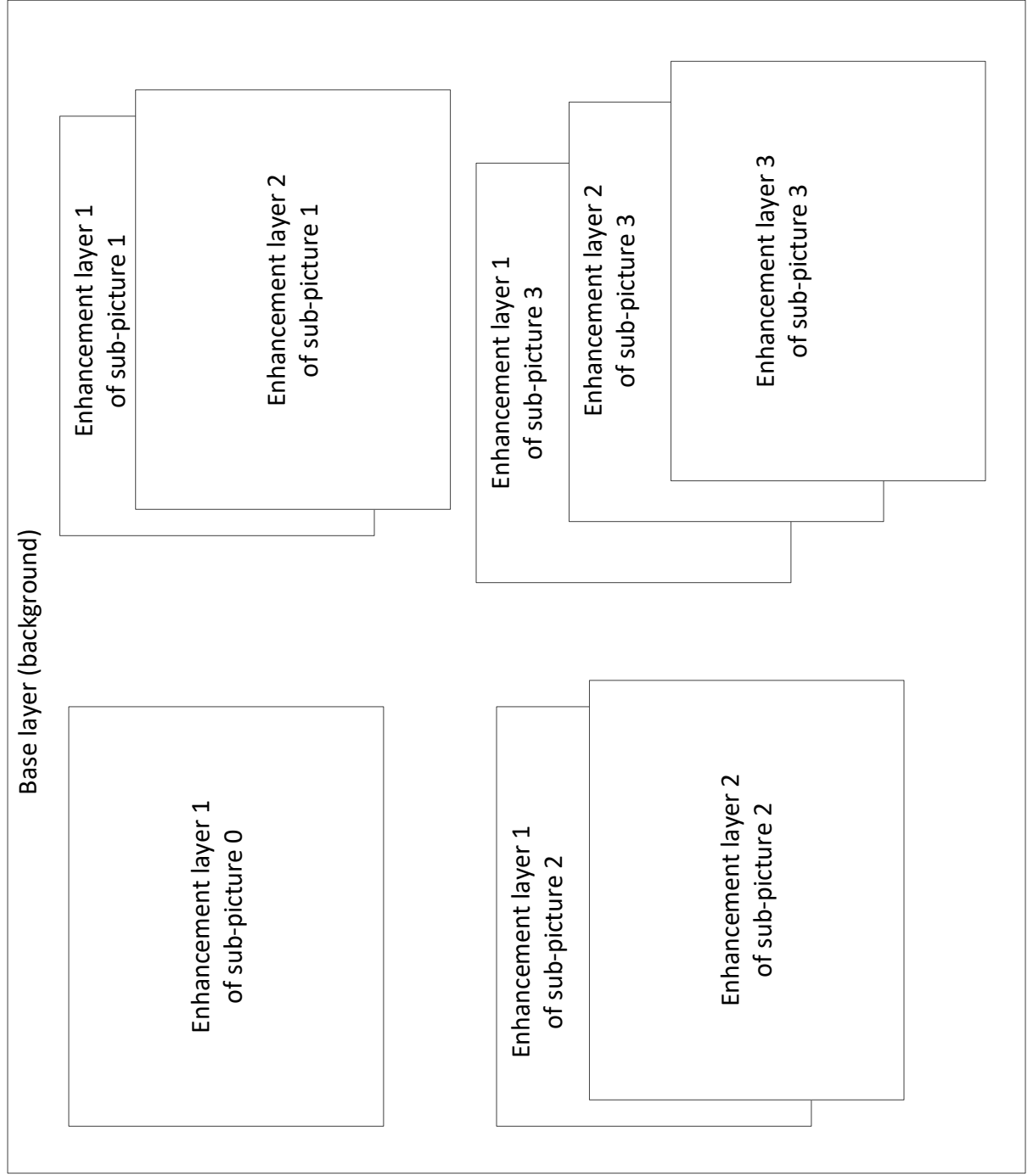


FIG. 11

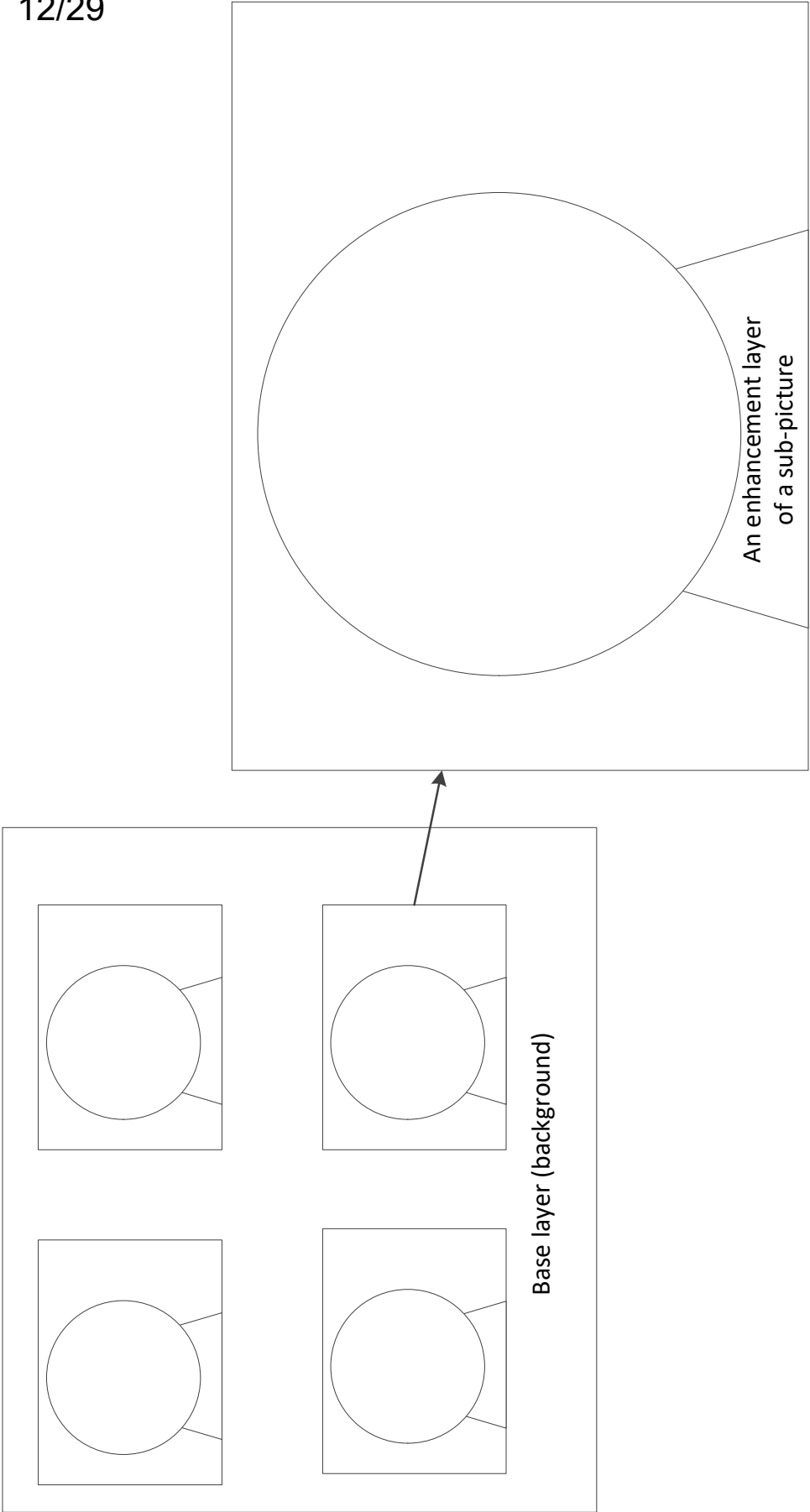


Fig. 12

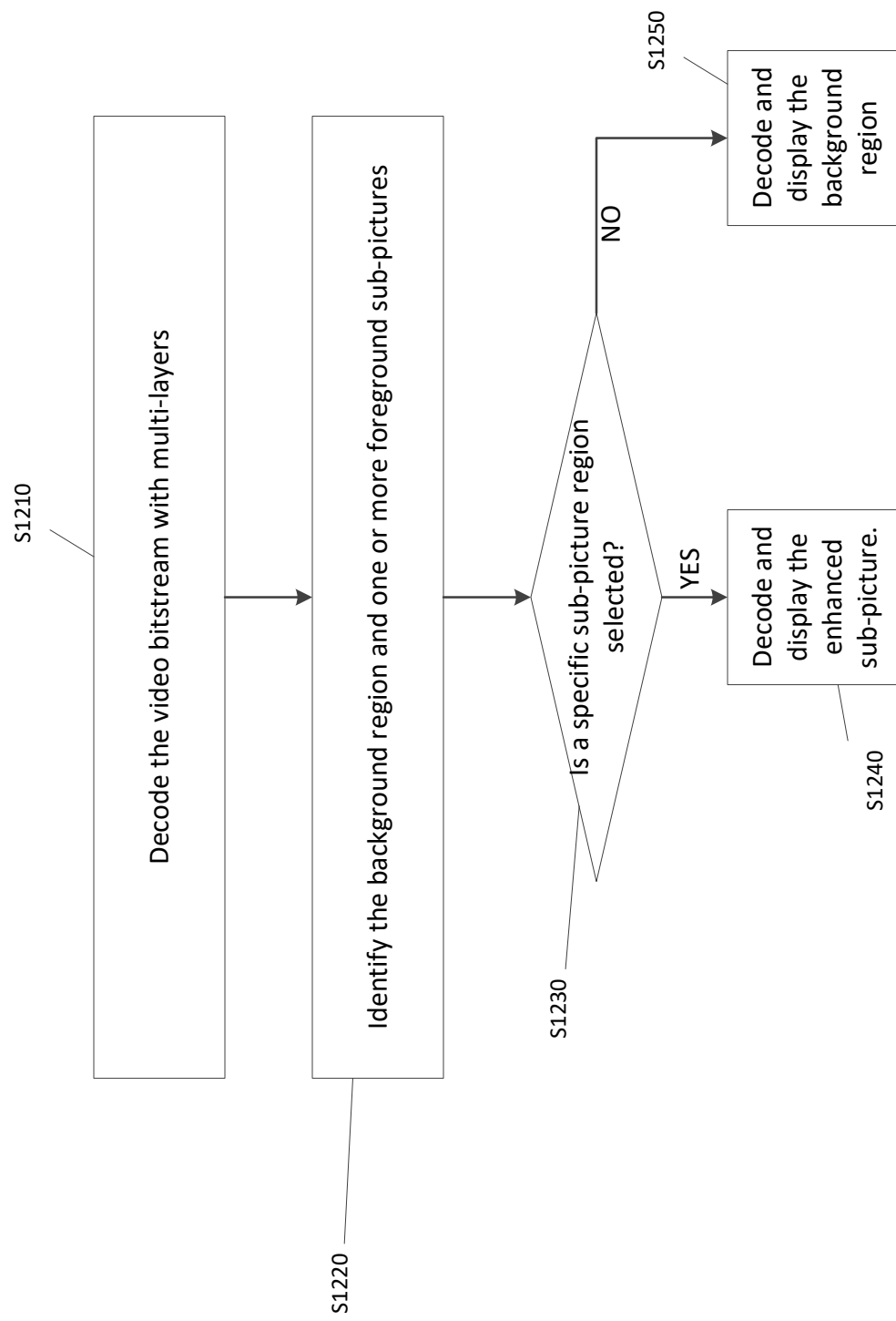


Fig. 13

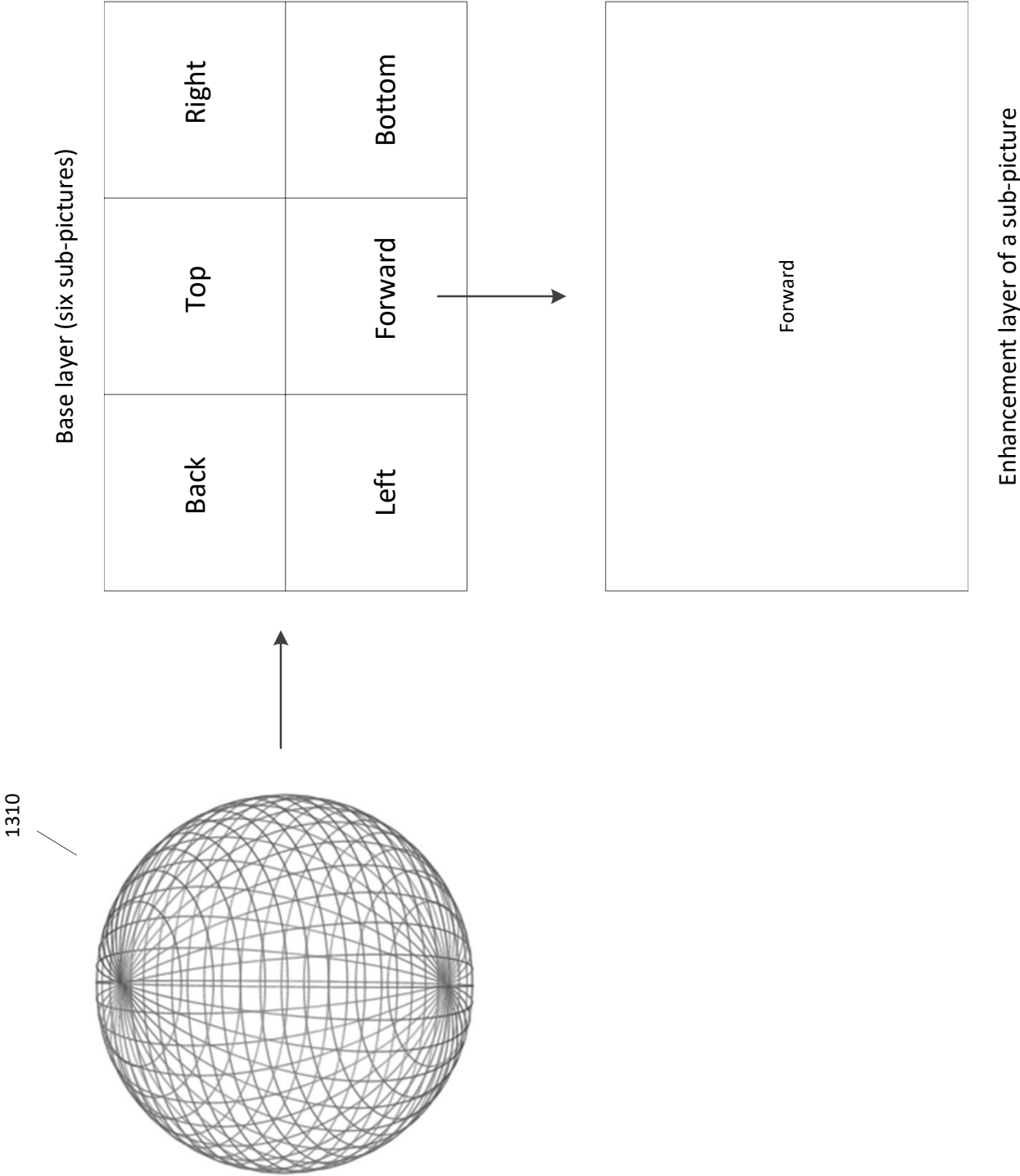


Fig. 14

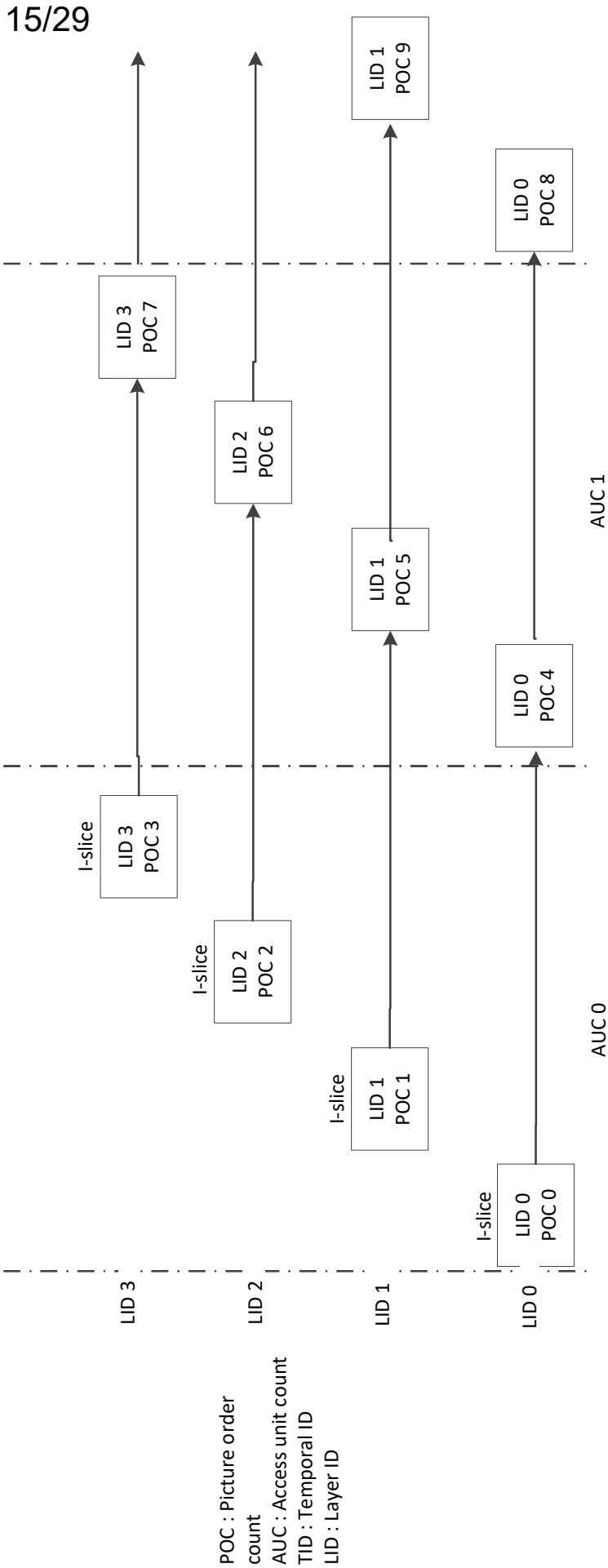
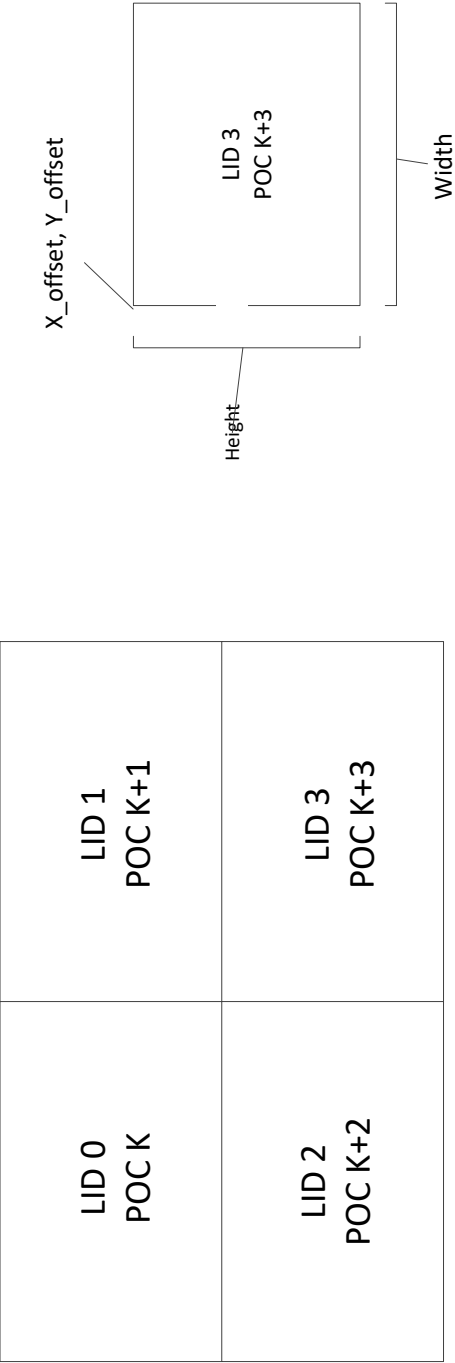
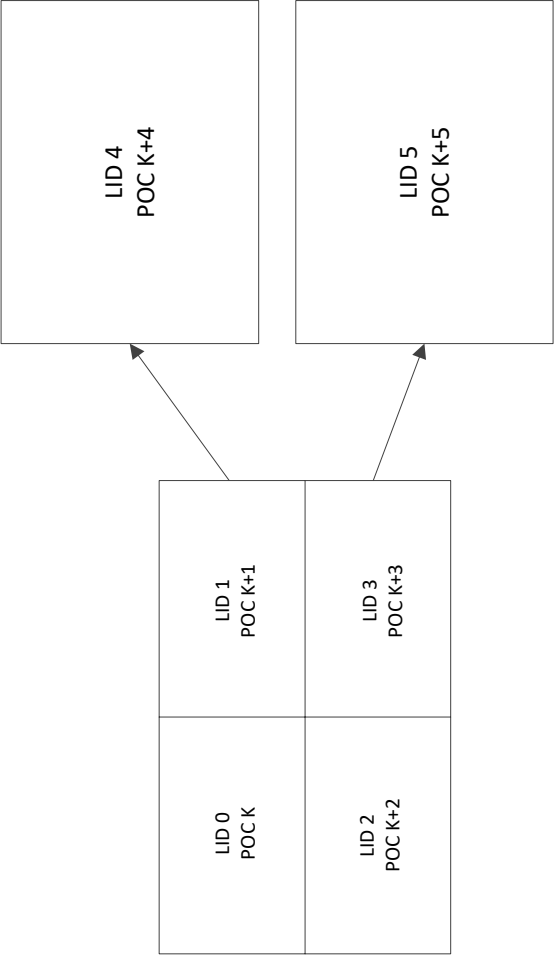
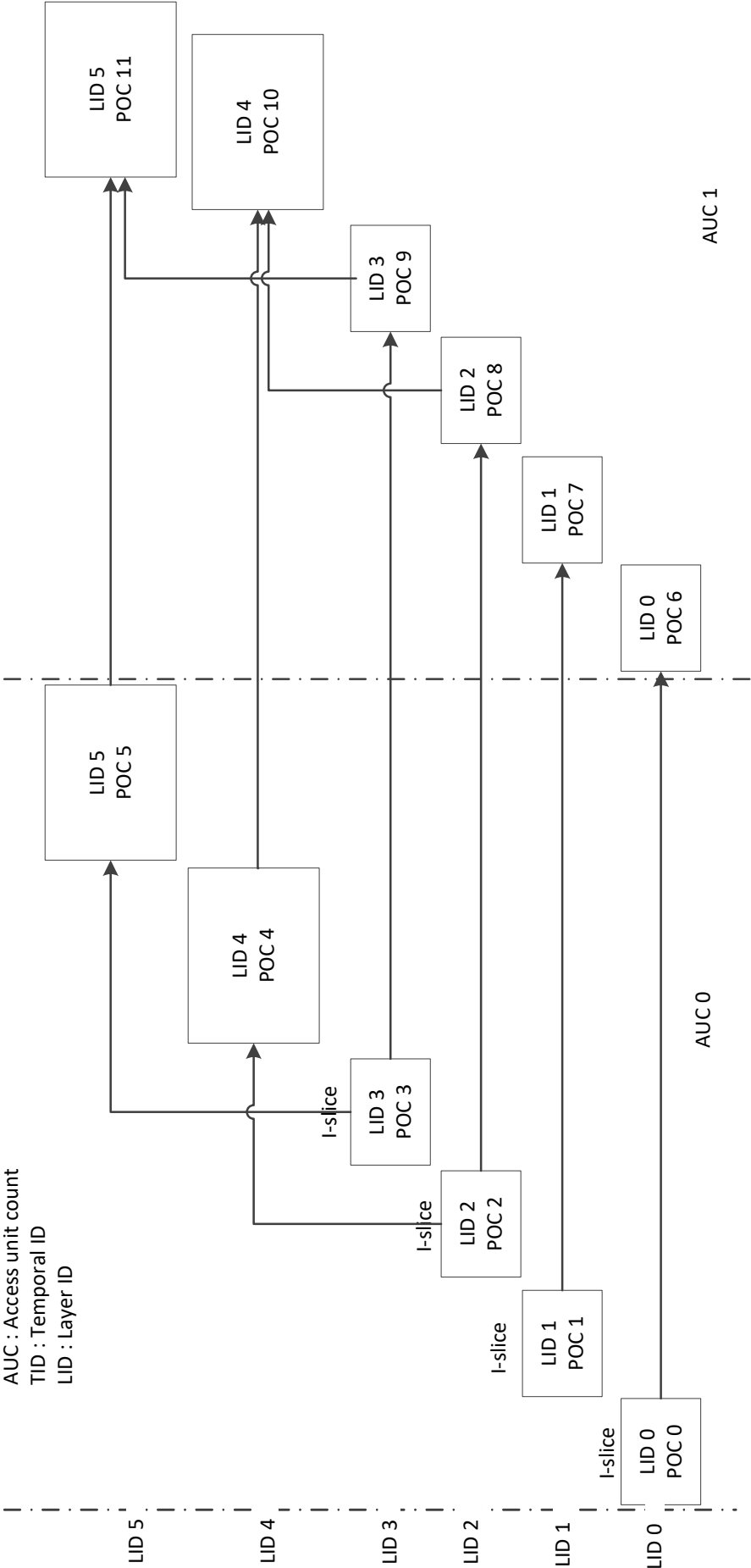


Fig. 15



POC : Picture order count
AUC : Access unit count
TID : Temporal ID
LID : Layer ID



	Descriptor
video_parameter_set_rbsp() {	
vps_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)
}	
...	
}	

FIG. 16A

	Descriptor
seq_parameter_set_rbsp() {	
sps_decoding_parameter_set_id	u(4)
sps_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. 16B

Fig. 17

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

Fig. 18

	Descriptor
video_parameter_set_rbsp() {	
...	
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 < NumLayersInList[i]; j++) {	
output_layer_flag[i][j]	u(1)
profile_tier_level_idx[i][j]	u(v)
}	
...	
}	

Fig. 19

	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)
}	
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 < NumLayersInList[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. 20

	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. 21

video_parameter_set_rbsp() {		
vps_video_parameter_set_id		u(4)
vps_max_layers_minus1		u(6)
vps_max_sublayers_minus1		u(3)
if (vps_max_layers_minus1 > 0 && vps_max_sublayers_minus1 > 0)		
vps_all_layers_same_num_sublayers_flag		u(1)
if (vps_max_layers_minus1 > 0)		
vps_all_independent_layers_flag		u(1)
for(i = 0; i <= vps_max_layers_minus1; i++) {		
vps_layer_id[i]		u(6)
if(i > 0 && !vps_all_independent_layers_flag) {		
vps_independent_layer_flag[i]		u(1)
if(!vps_independent_layer_flag[i]) {		
for(j = 0; j < i; j++)		
vps_direct_ref_layer_flag[i][j]		u(1)
max_tid_ref_present_flag[i]		u(1)
if(max_tid_ref_present_flag[i])		
max_tid_ref_pics_plus1[i]		u(3)
}		
}		
}		
if (vps_max_layers_minus1 > 0) {		
if (vps_all_independent_layers_flag)		
each_layer_is_an_ols_flag		u(1)
if (each_layer_is_an_ols_flag) {		
if (vps_all_independent_layers_flag)		
ols_mode_idc		u(2)
if (ols_mode_idc == 2) {		
num_output_layer_sets_minus1		u(8)
for(i = 1; i <= num_output_layer_sets_minus1; i++)		
for(j = 0; j <= vps_max_layers_minus1; j++)		
ols_output_layer_flag[i][j]		u(1)
}		
}		
}		
}		
}		
}		

Fig. 22

	Descriptor
video_parameter set rbsp() {	
vps_video_parameter set id	u(4)
vps_max_layers_minus1	u(6)
vps_max_sublayers_minus1	u(3)
if(vps_max_layers_minus1 > 0 && vps_max_sublayers_minus1 > 0)	
vps_all_layers_same_num_sublayers_flag	u(1)
if(vps_max_layers_minus1 > 0)	
vps_all_independent_layers_flag	u(1)
for(i = 0; i <= vps_max_layers_minus1; i++) {	
vps_layer_id[i]	u(6)
if(i > 0 && !vps_all_independent_layers_flag) {	
vps_independent_layer_flag[i]	u(1)
if(!vps_independent_layer_flag[i]) {	
for(j = 0; j < i; j++)	
vps_direct_ref_layer_flag[i][j]	u(1)
max_tid_ref_present_flag[i]	u(1)
if(max_tid_ref_present_flag[i])	
max_tid_il_ref_pics_plus1[i]	u(3)
}	
}	
if(vps_max_layers_minus1 > 0) {	
if(vps_all_independent_layers_flag)	
each_layer_is_an_ols_flag	u(1)
if(!each_layer_is_an_ols_flag) {	
if(!vps_all_independent_layers_flag)	
ols_mode_idc	u(2)
if(ols_mode_idc == 2 && vps_max_layers_minus1 > 1) {	
num_output_layer_sets_minus1	u(8)
for(i = 1; i <= num_output_layer_sets_minus1; i++)	
for(j = 0; j <= vps_max_layers_minus1; j++)	
ols_output_layer_flag[i][j]	u(1)
}	
}	
}	
}	
}	
}	
...	
}	

Fig. 23

seq parameter set rbsp() {	Descriptor
...	
sps_ref_pic_resampling_enabled_flag	u(1)
if(sps_ref_pic_resampling_enabled_flag)	
sps_res_change_in_clvs_allowed_flag	u(1)
...	
if(!sps_res_change_in_clvs_allowed_flag)	
sps_subpic_info_present_flag	u(1)
if(sps_subpic_info_present_flag) {	
...	
}	
...	
sps_virtual_boundaries_enabled_flag	u(1)
if(sps_virtual_boundaries_enabled_flag) {	
if(!sps_res_change_in_clvs_allowed_flag)	
sps_virtual_boundaries_present_flag	u(1)
if(sps_virtual_boundaries_present_flag) {	
...	
}	
}	
...	
}	

Fig. 24

pic_parameter_set_rbsp() {	Descriptor
...	
pps_res_change_in_clvs_allowed_flag	u(1)
if(pps_res_change_in_clvs_allowed_flag) {	
pps_pic_width_in_luma_samples	ue(v)
pps_pic_height_in_luma_samples	ue(v)
}	
...	
}	

FIG. 25A

2500A

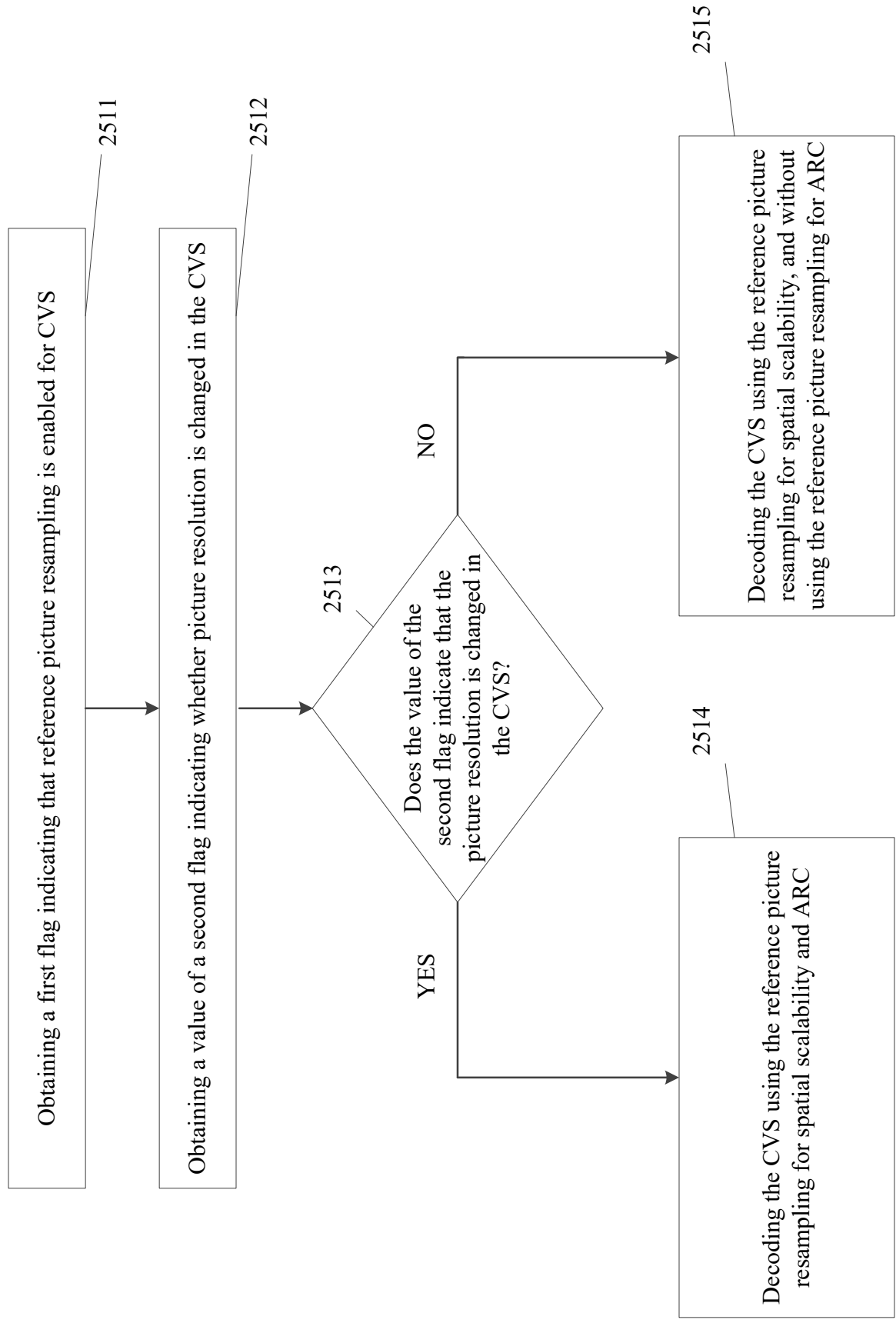


FIG. 25B 2500B

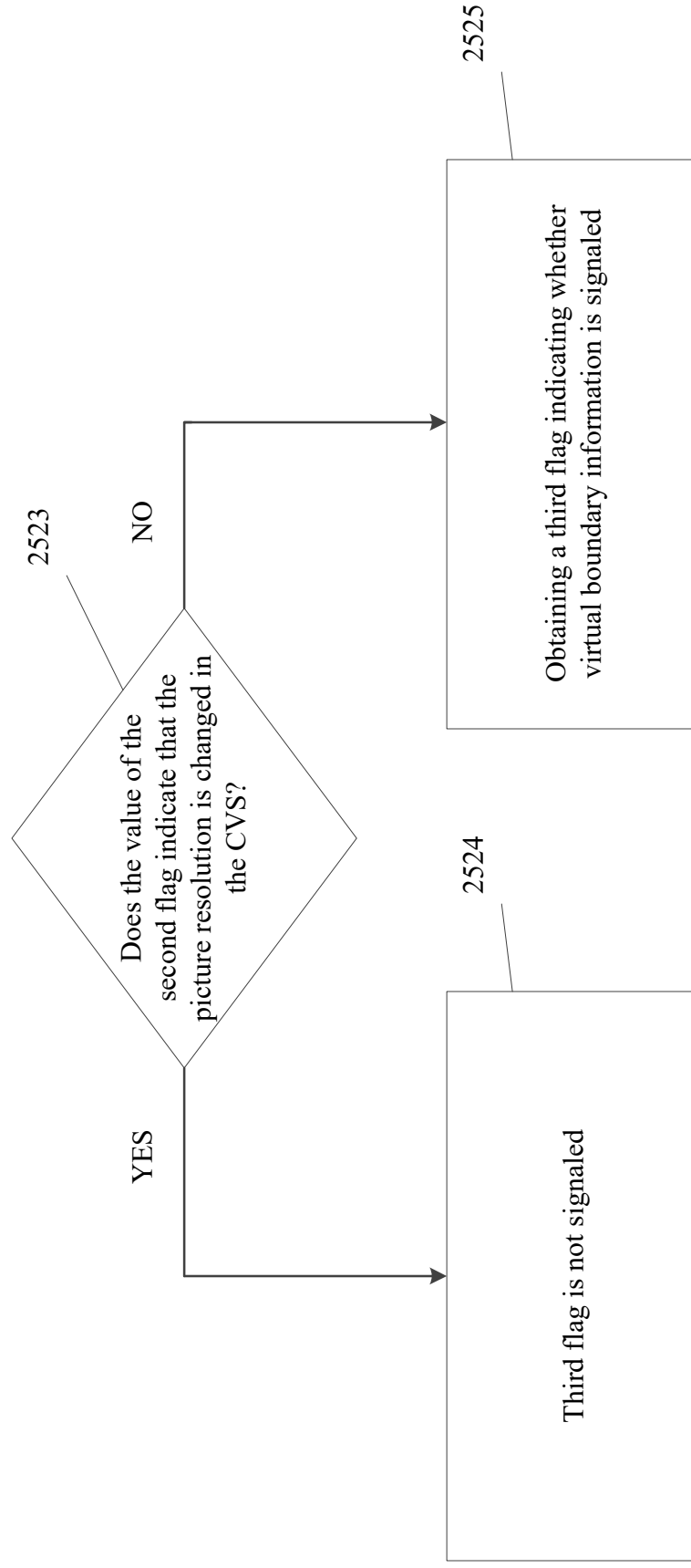


FIG. 25C 2500C

