

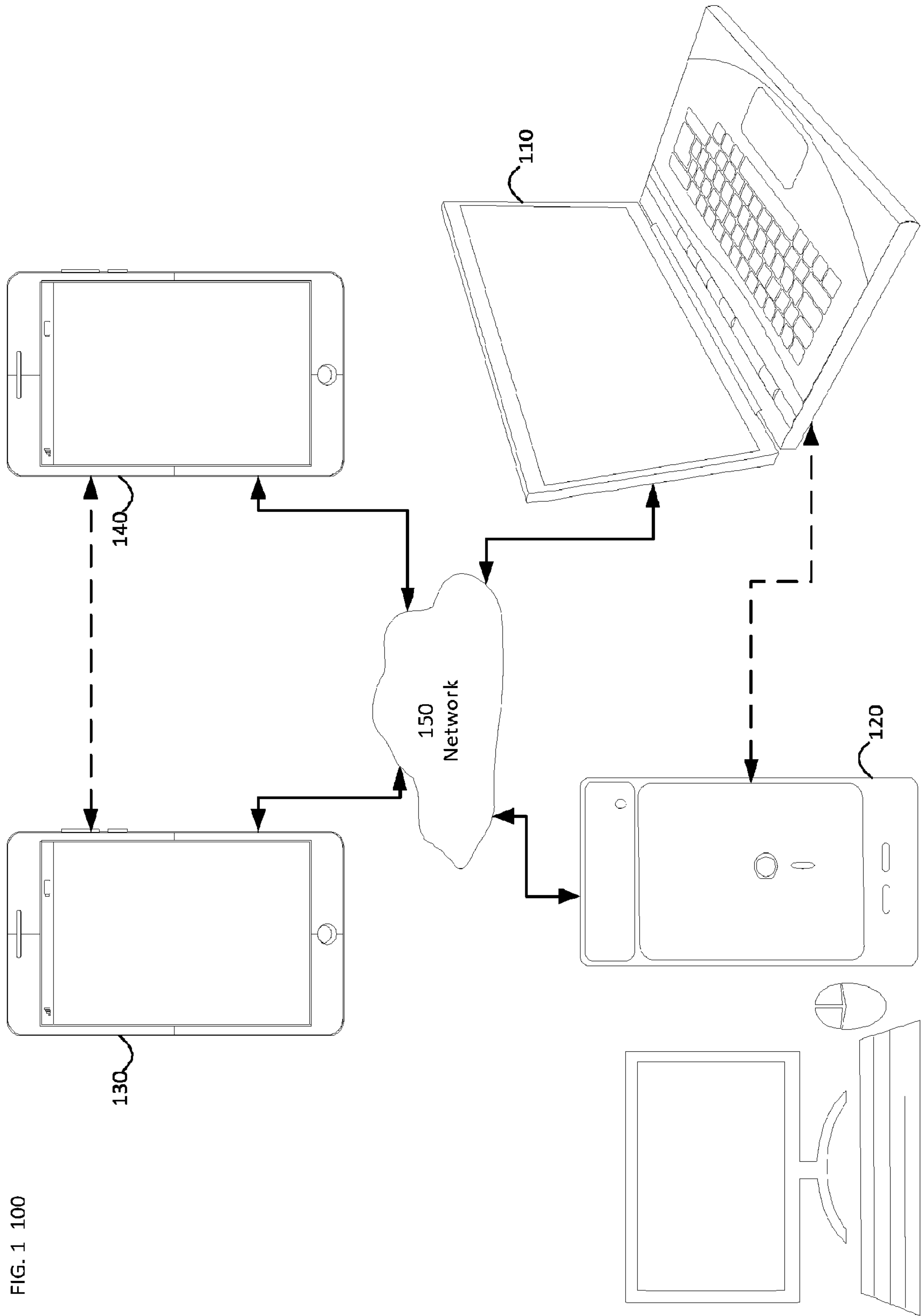


FIG. 1 100

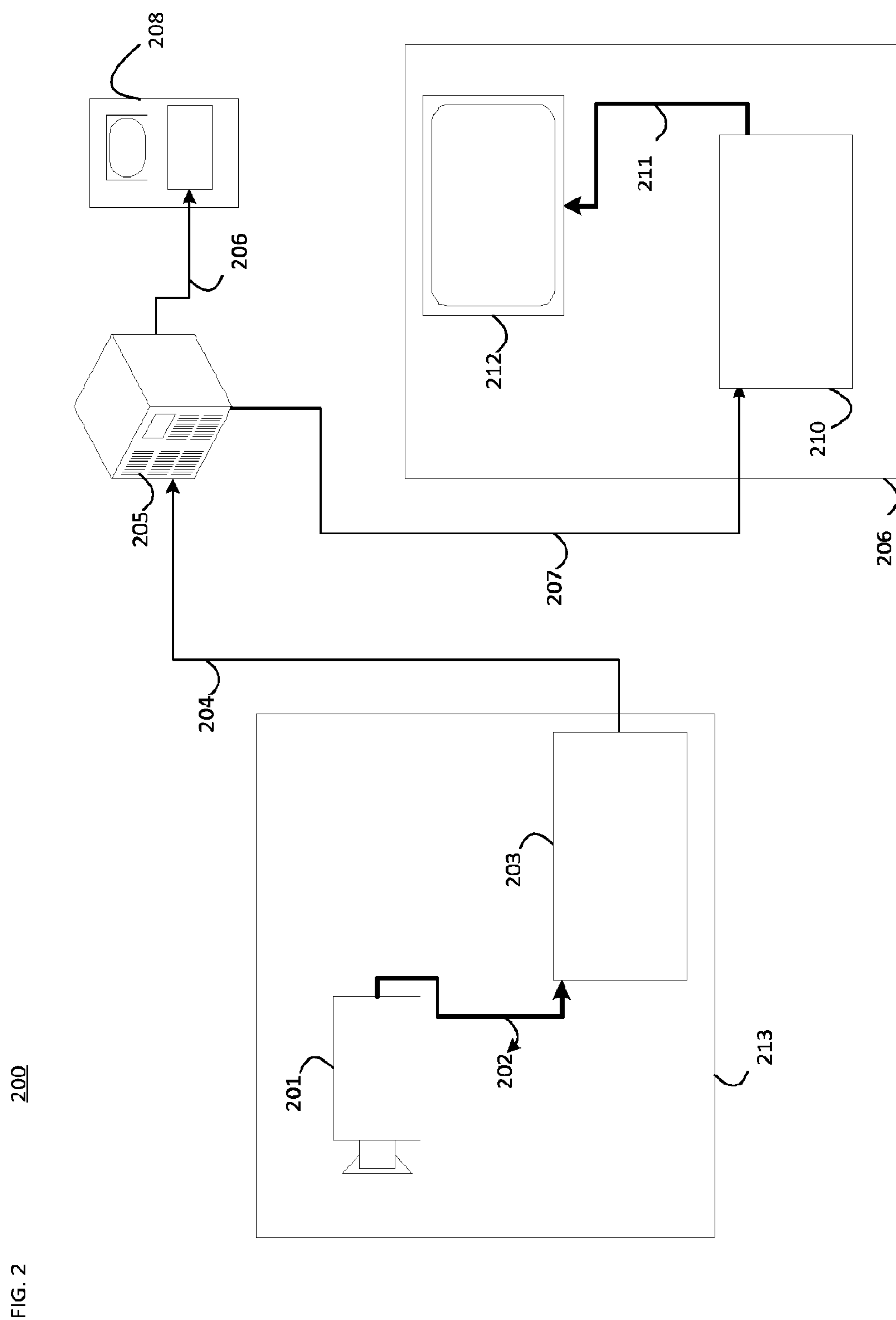


FIG. 3 Decoder 210

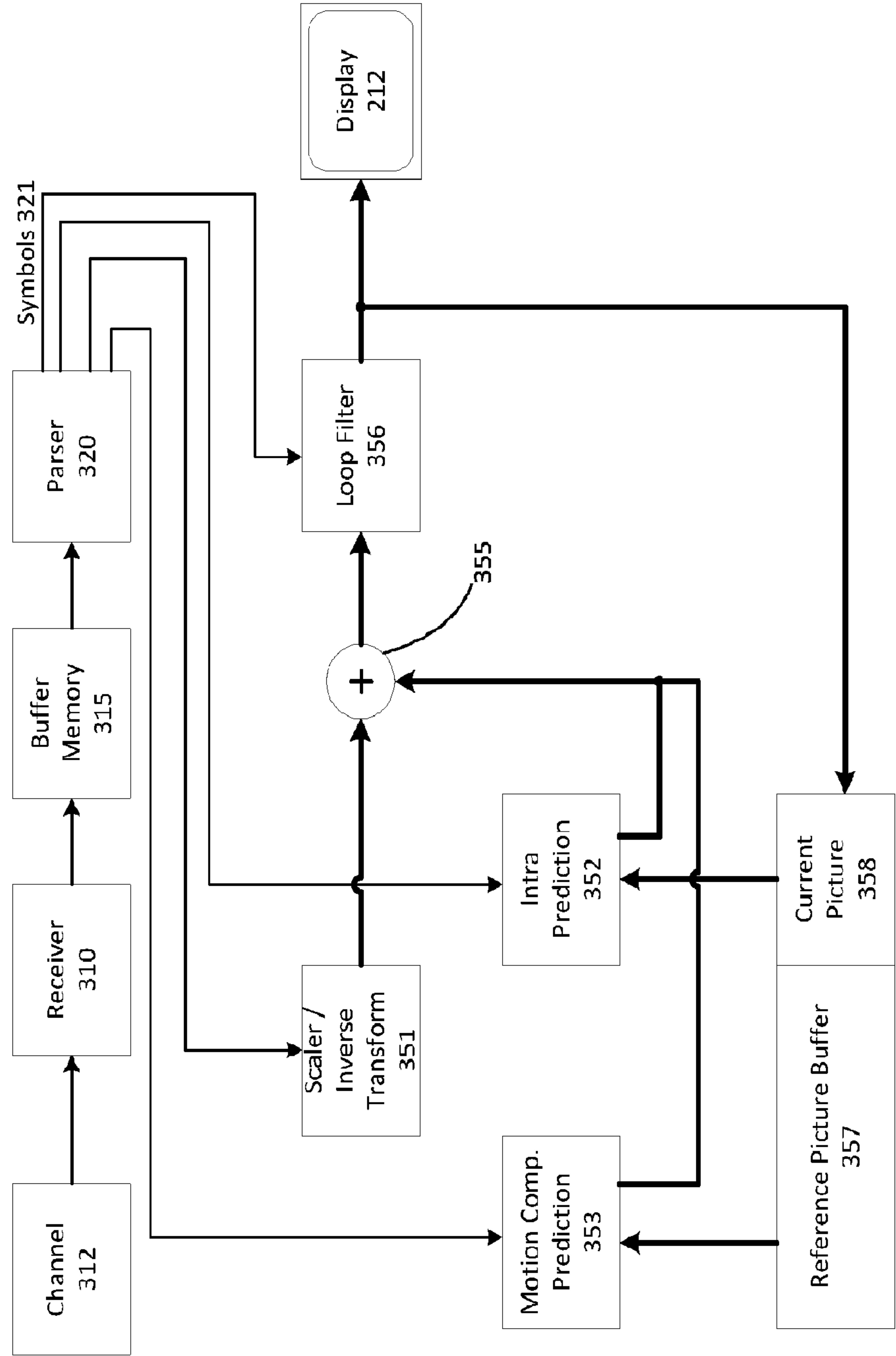
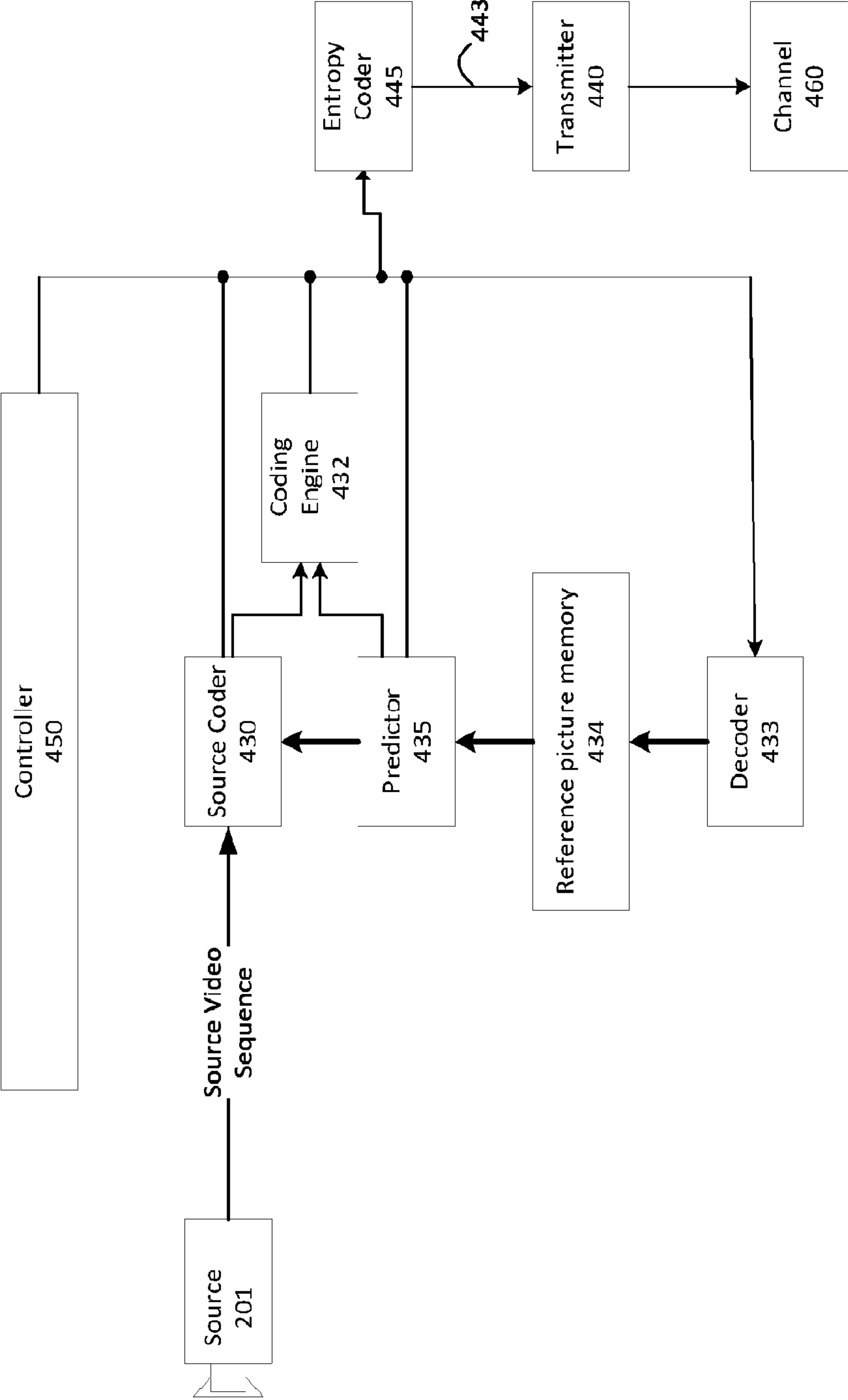


FIG. 4 Encoder 203



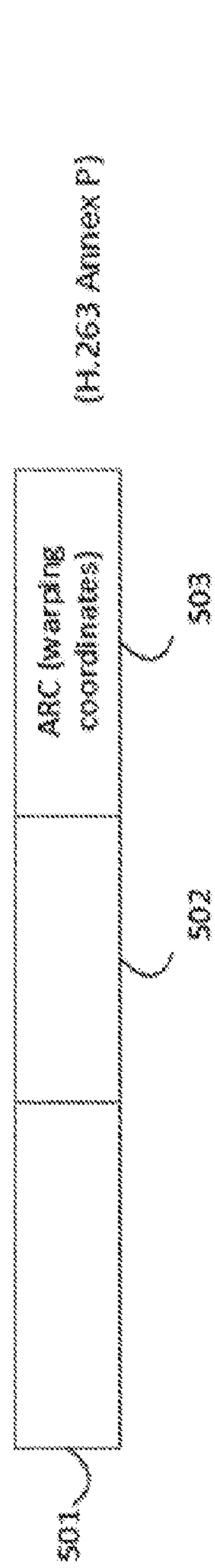


FIG. 5A

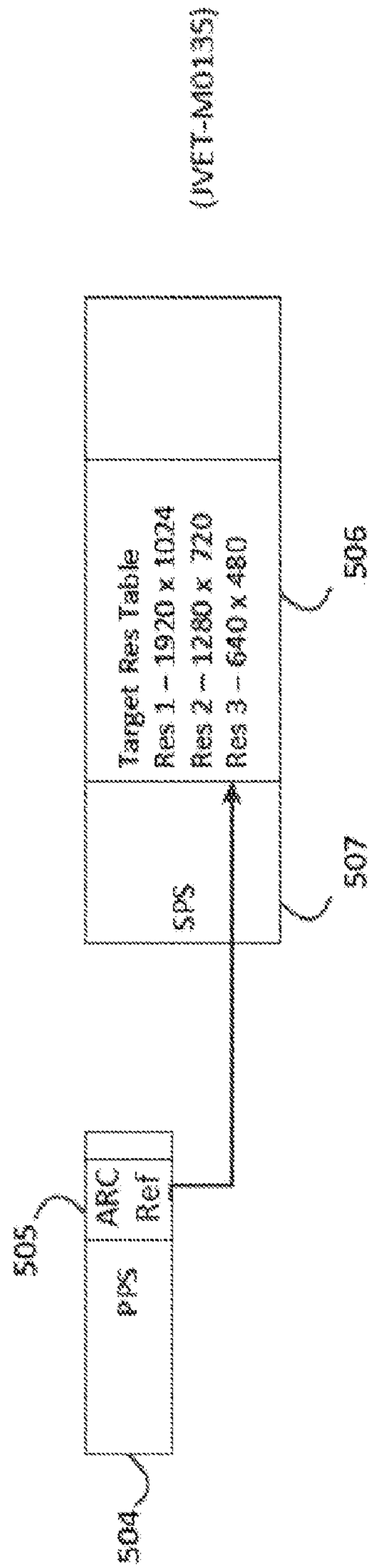


FIG. 5B

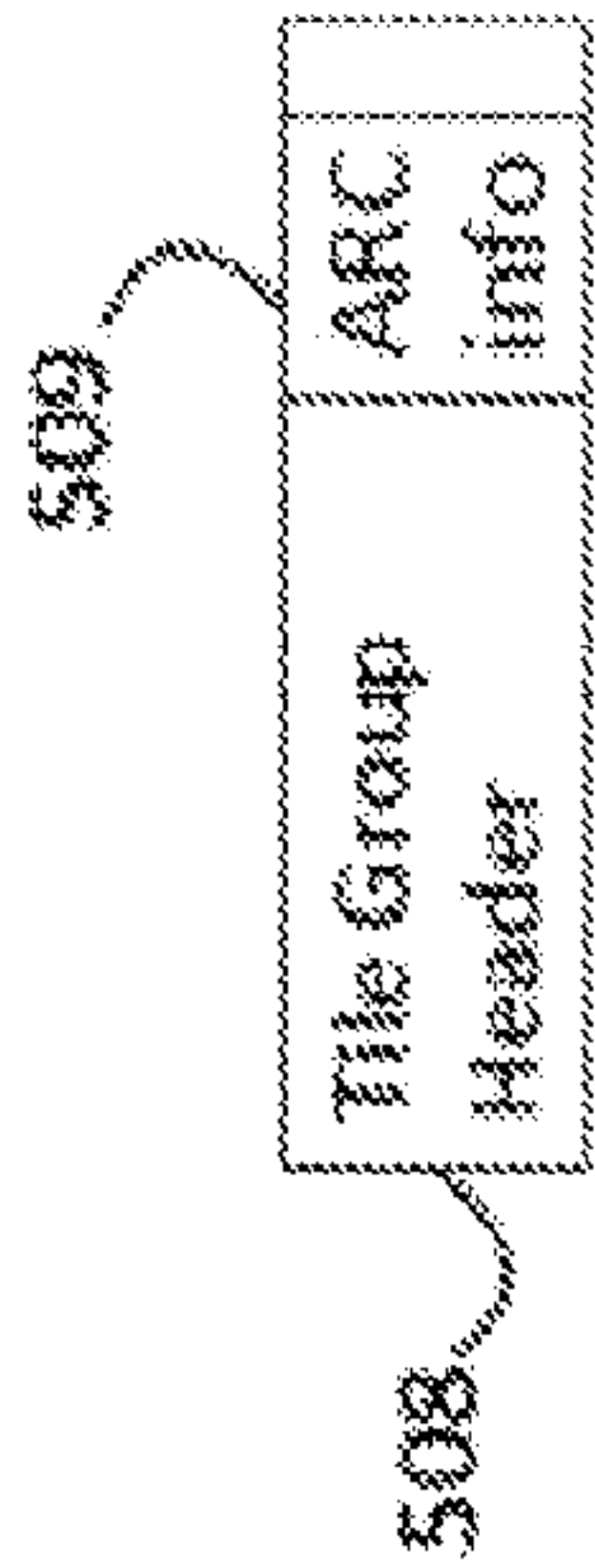


FIG. 5C

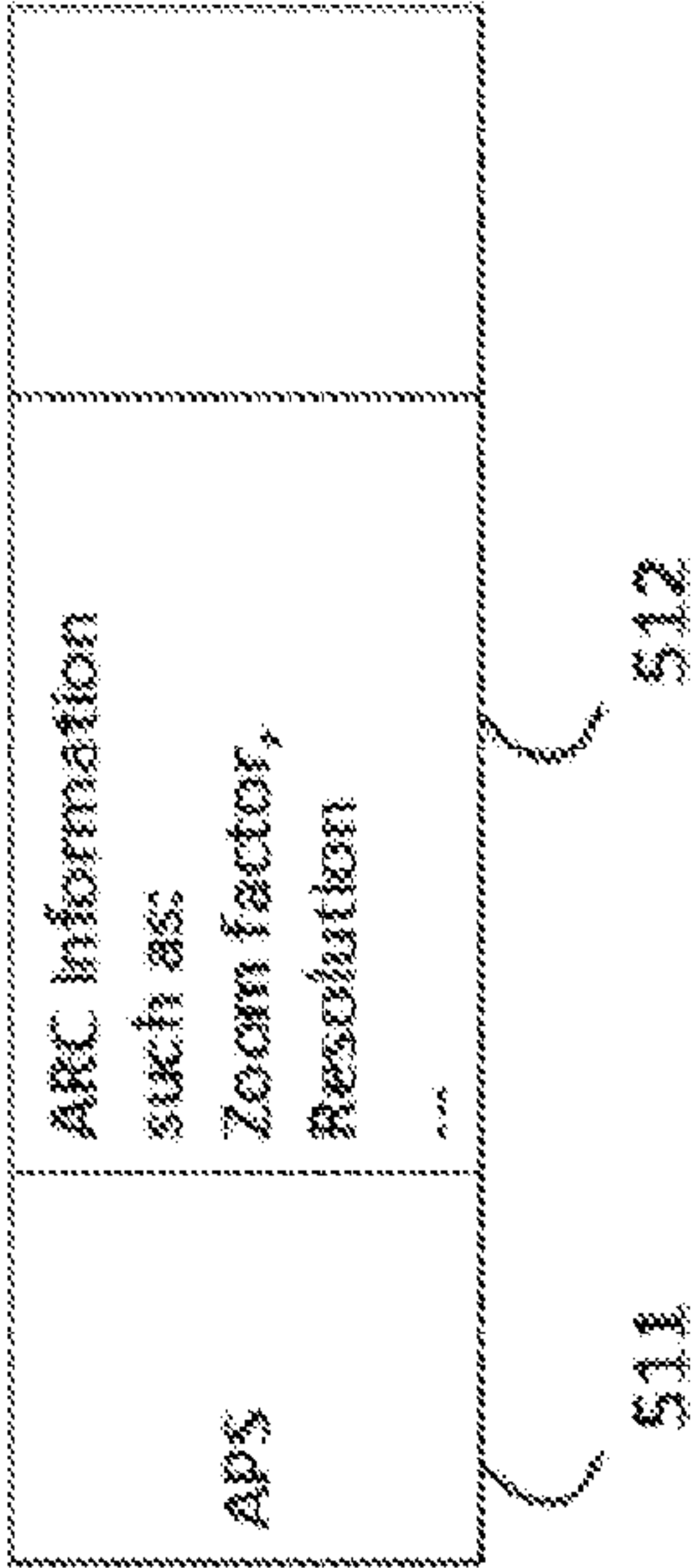


FIG. 5D

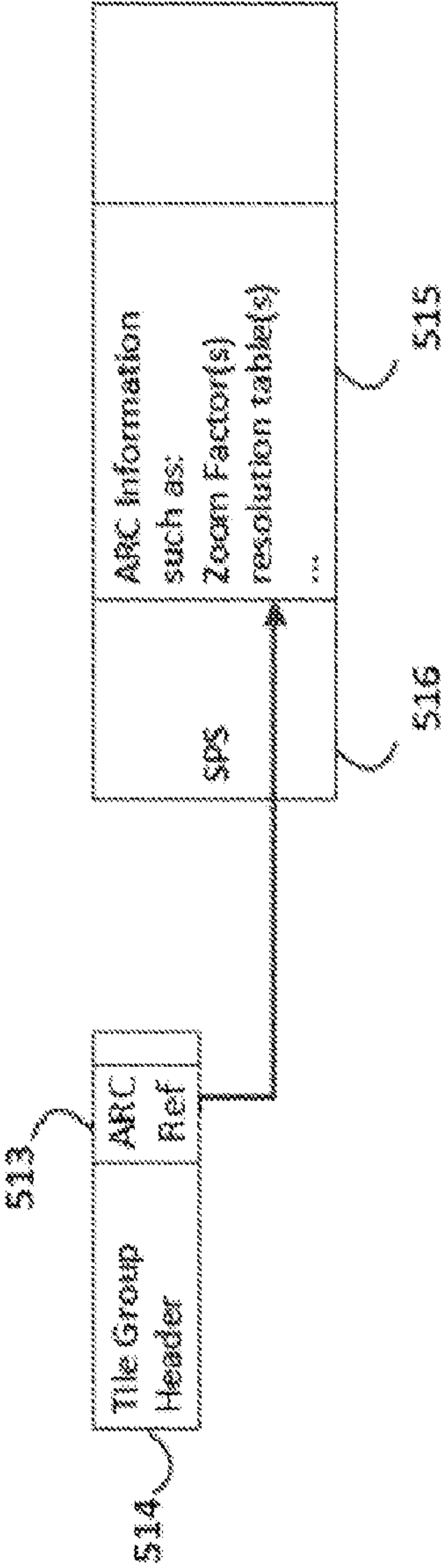


FIG. 5E

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++) {	
	dec_pic_width_in_luma_samples[i]	ue(v)
617	dec_pic_height_in_luma_samples[i]	ue(v)
	}	
	}	
	...	
	}	

FIG. 6B

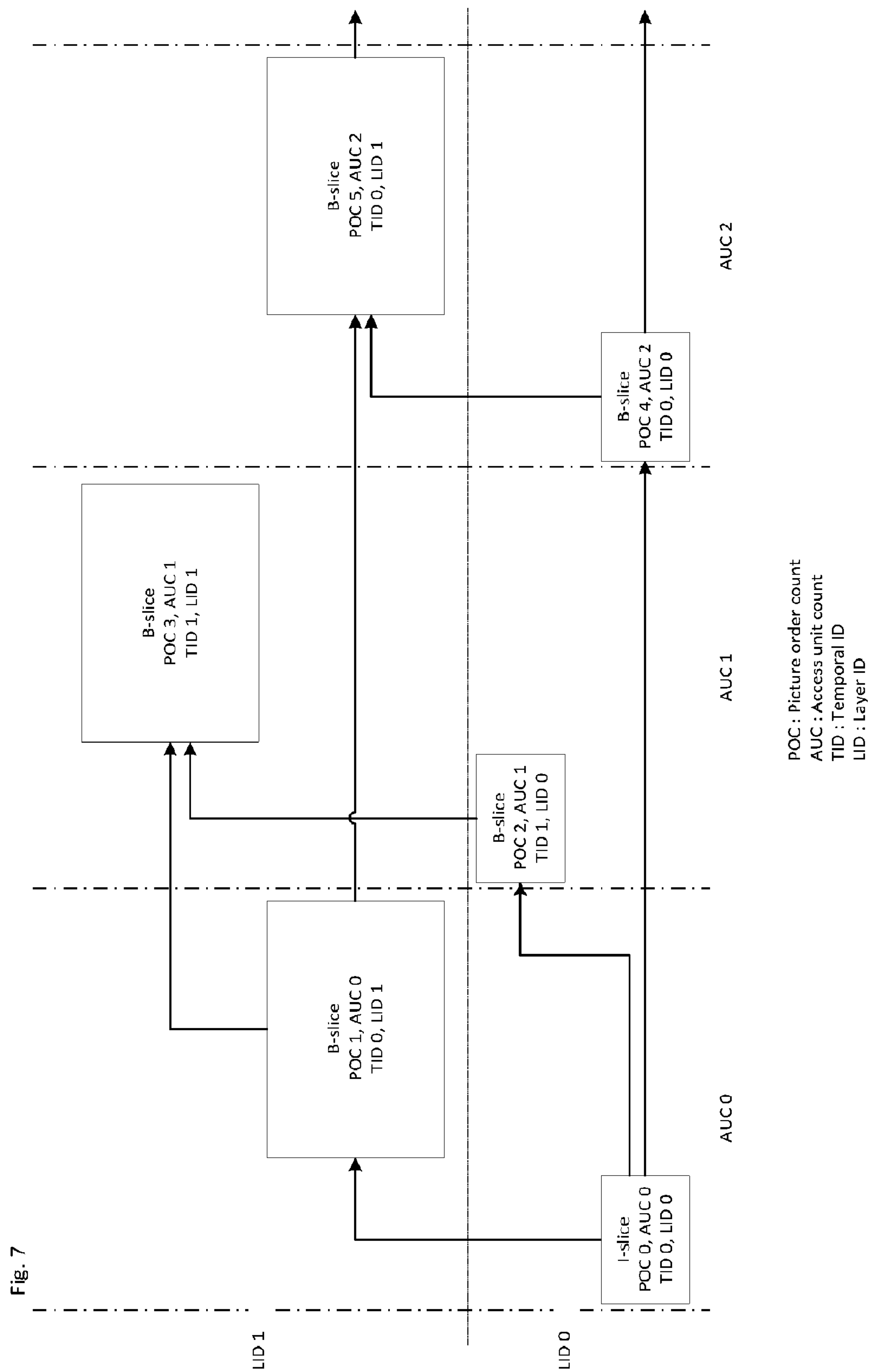


Fig. 8

video_parameter_set_header() { vps_video_parameter_set_id vps_max_layers_minus1 for(i = 0; i <= vps_max_layers_minus1; i++) { vps_included_layer_id[i] vps_reserved_zero_bit } vps_constraint_info_present_flag vps_constant_poc_cycle_per_aub if(vps_constant_poc_cycle_per_aub) vps_poc_cycle_aub ... }	Descriptor u(4) u(8) u u(7) u(1) u u(1) u(1) u(1) u(3) u u
slice_header() { slice_pic_parameter_set_id if(rect_slice_flag == NonRectInPic) { slice_address } if(rect_slice_flag && (angle_brck_per_slice_flag) num_bricks_in_slice_minus1 slice_type if(NalUnitType == GRA_NUT) { recovery_poc_cnt } slice_pic_order_lsb ... if(vps_constant_poc_cycle_per_aub) slice_poc_cycle_aub ... }	Descriptor ue(v) u u(v) u ue(v) ue(v) u ue(v) u(v) u u(3) u u u

Fig. 9

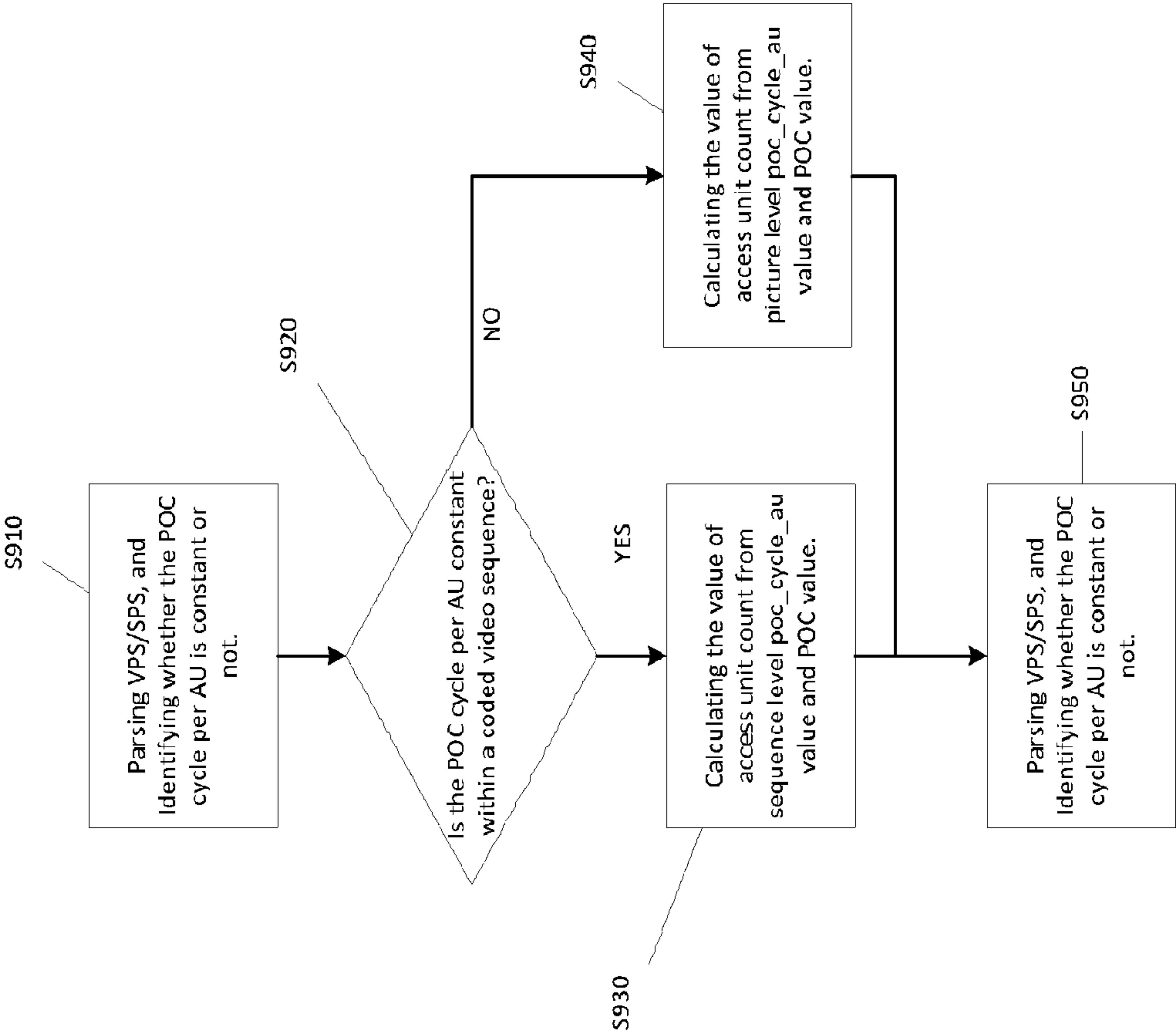


FIG. 10

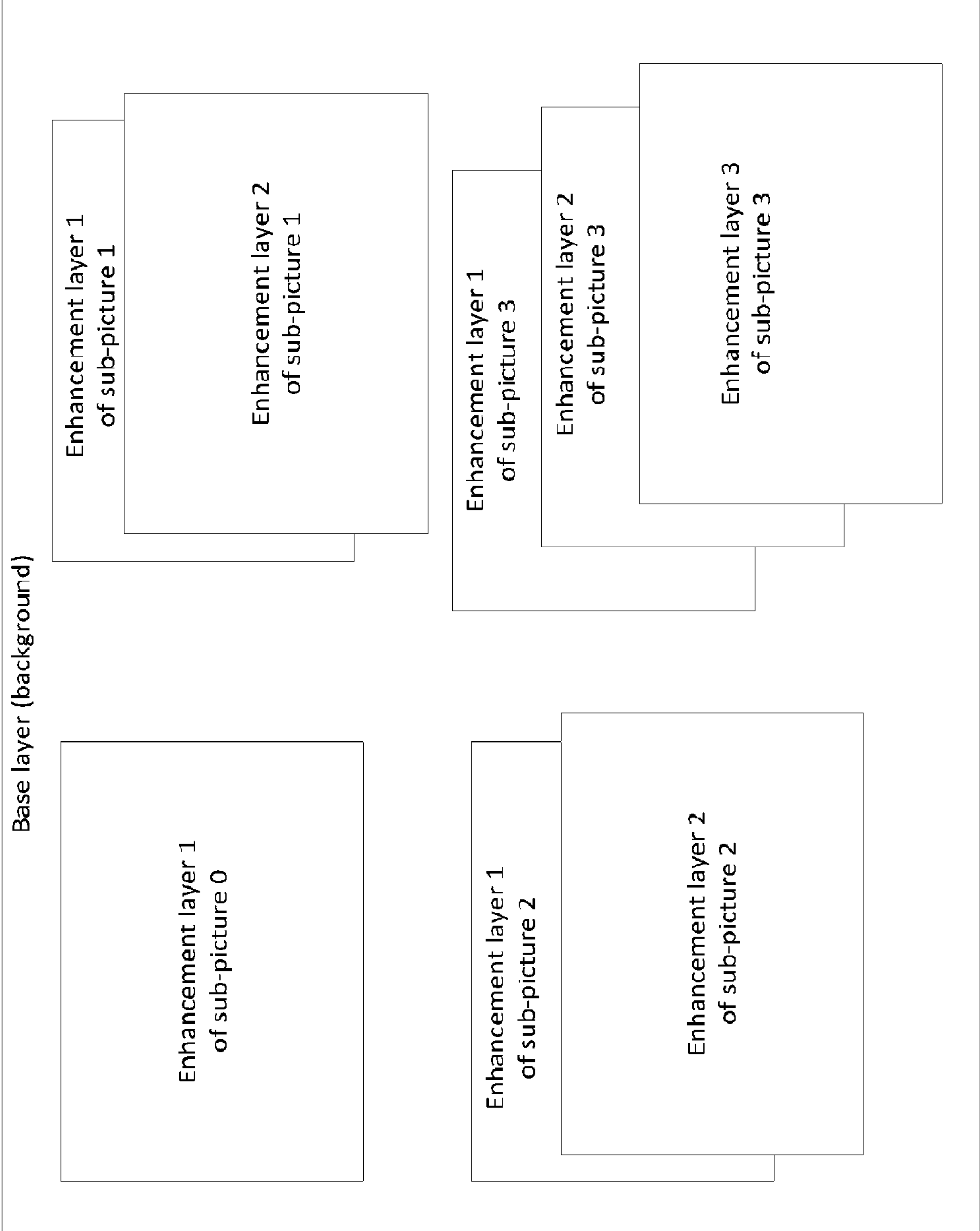


FIG. 11

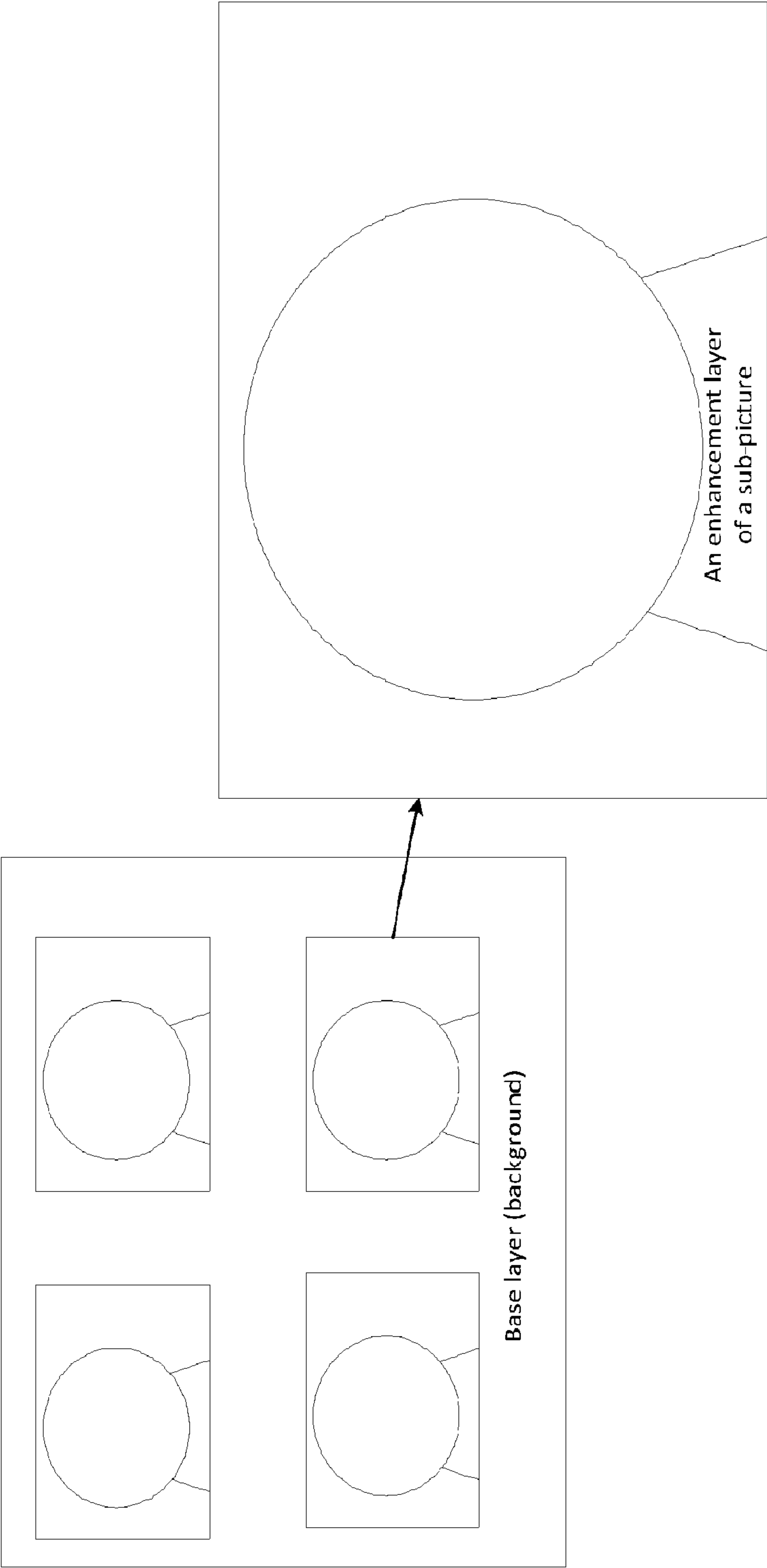


Fig. 12

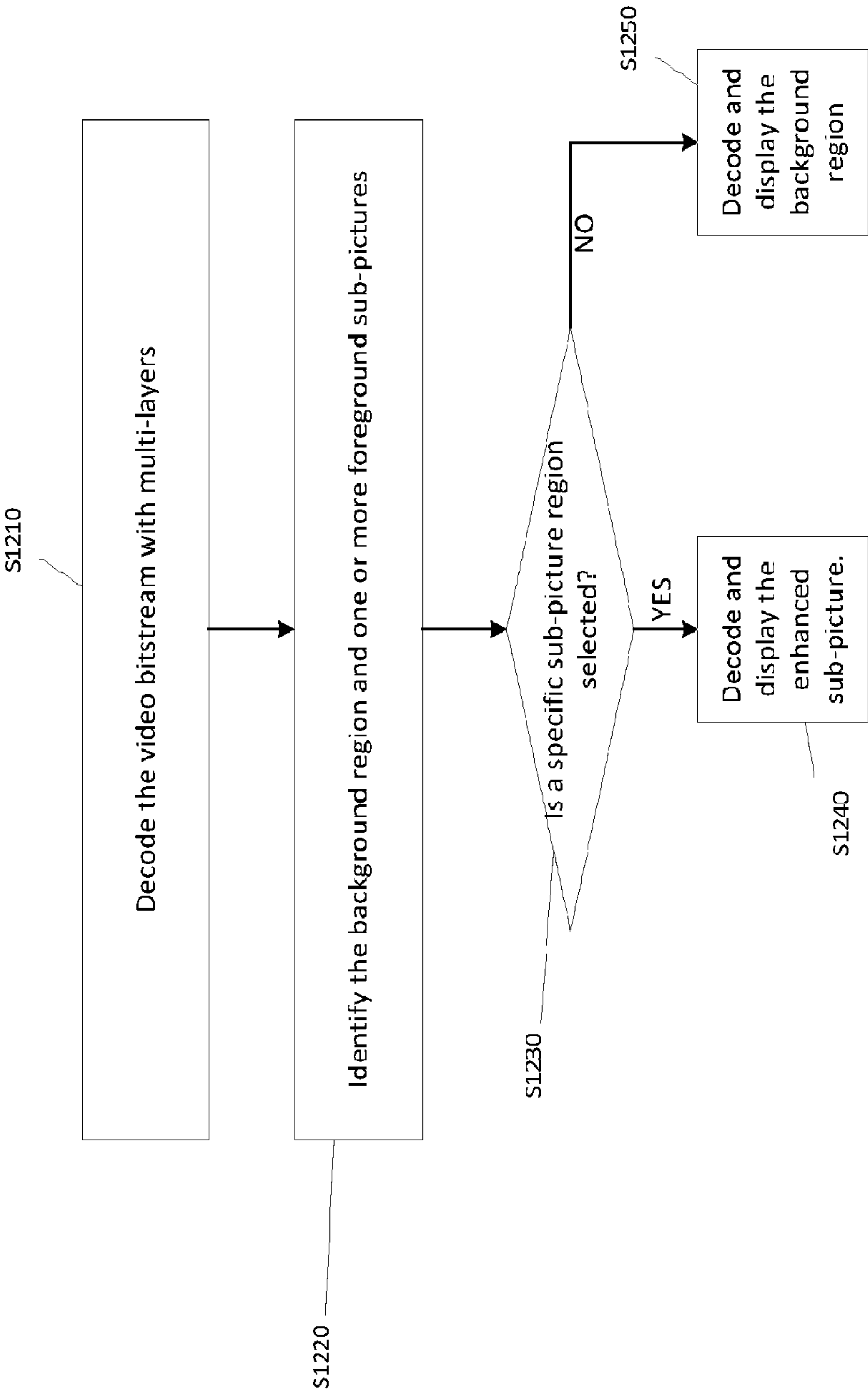


Fig. 13

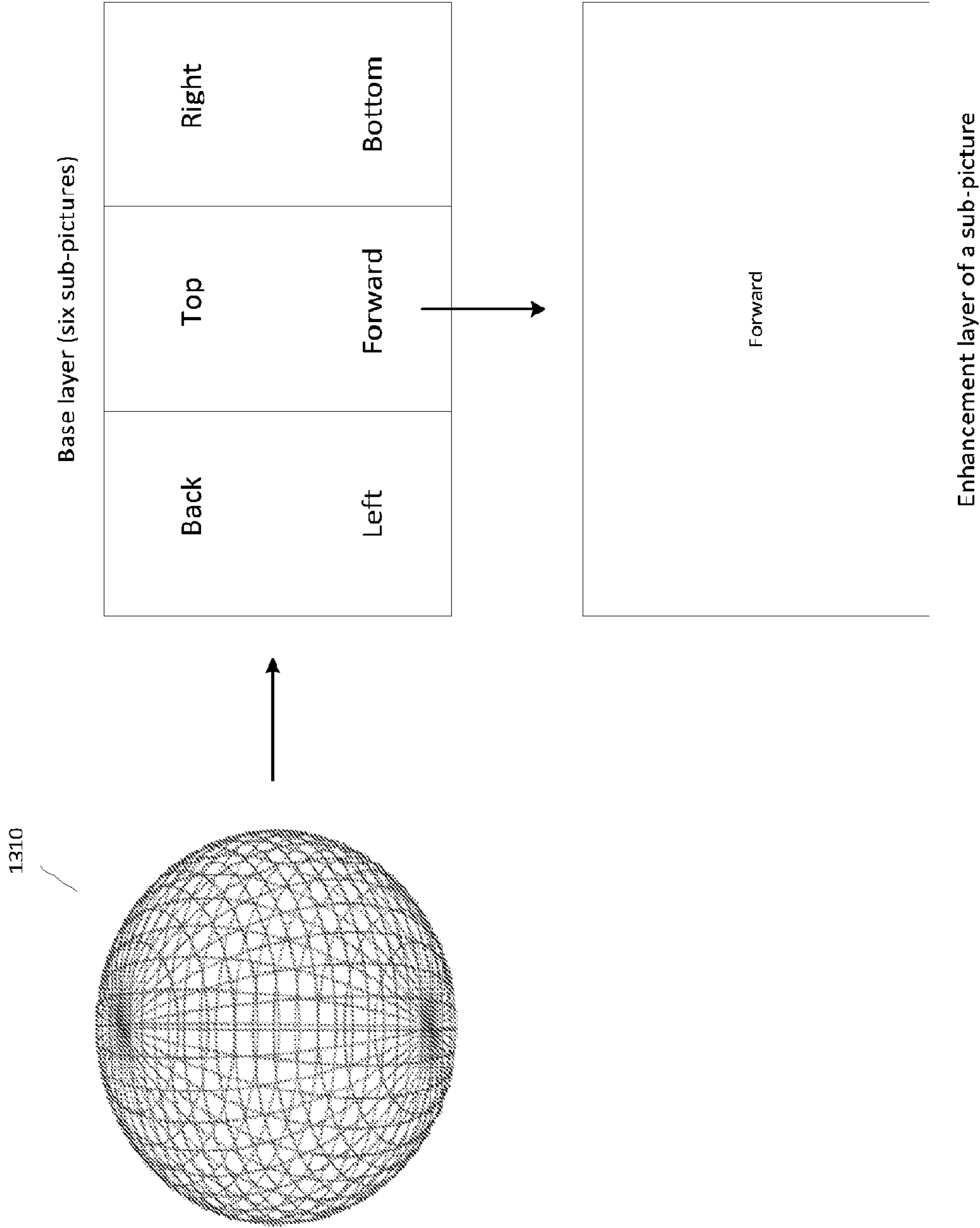
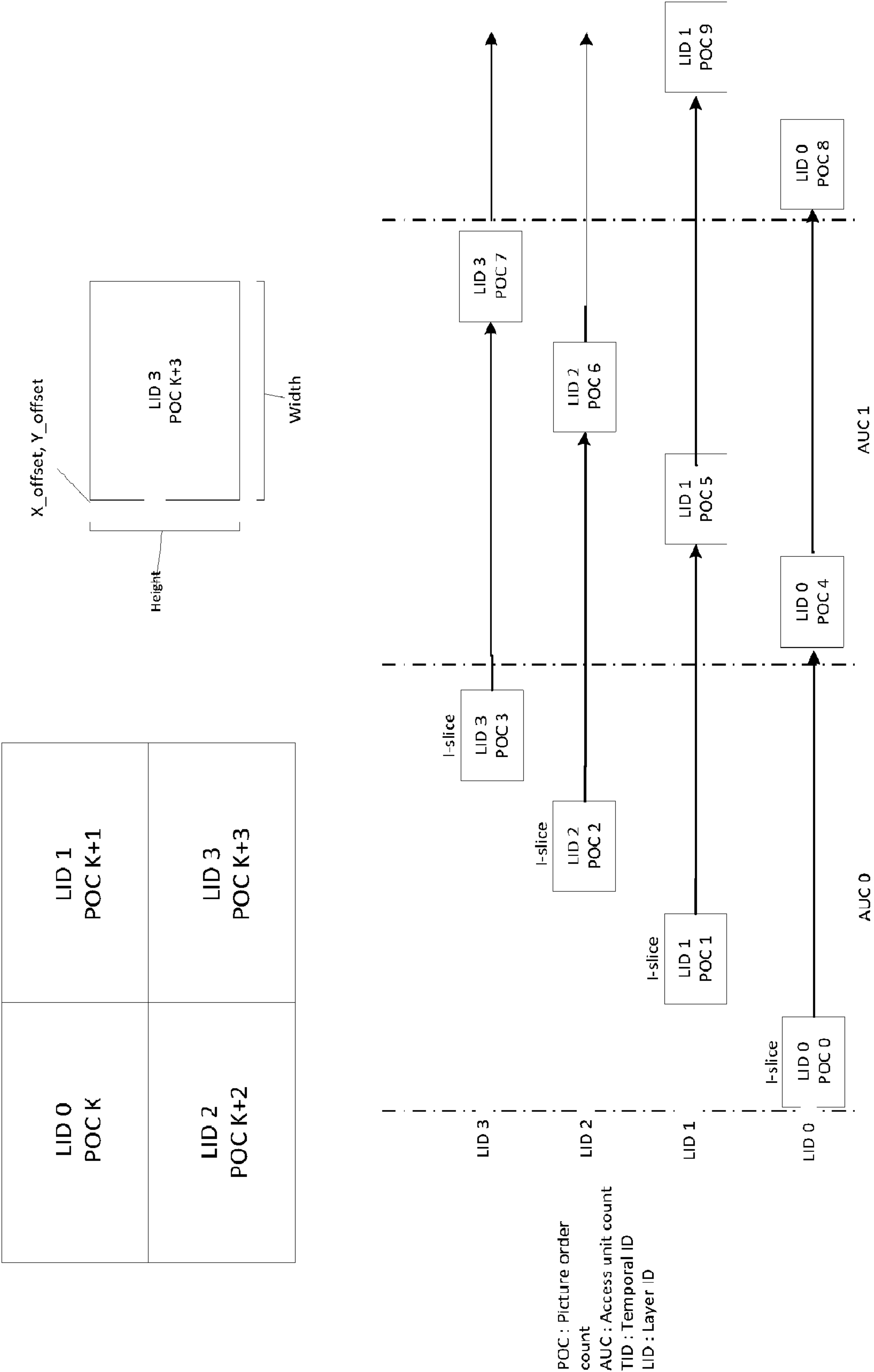
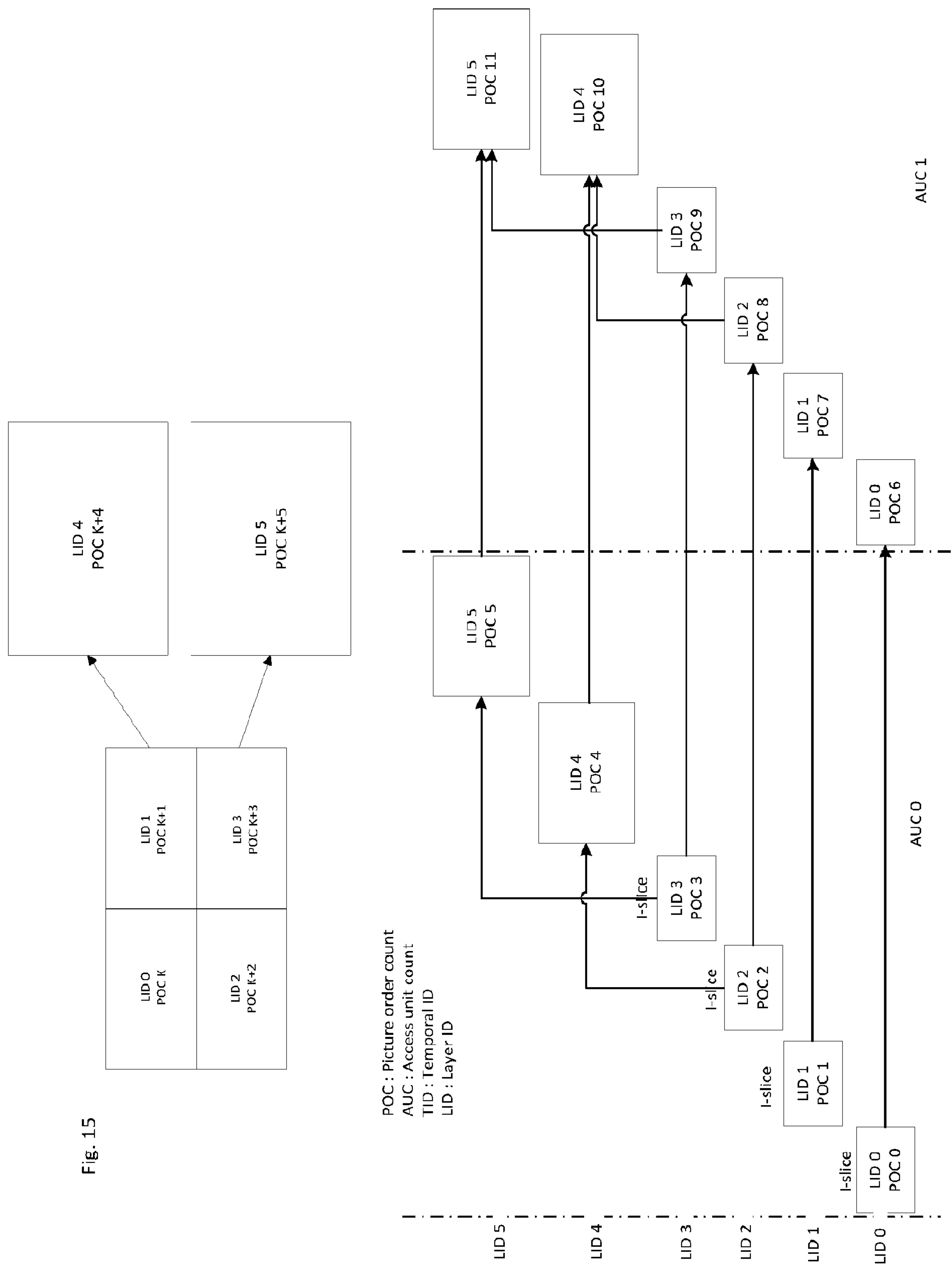


Fig. 14





video_parameter_set_id; { sps_video_parameter_set_id; vps_max_layers_minus1; for(i = 0; i <= vps_max_layers_minus1; i++) { vps_included_layer_id[i]; vps_reserved_zero_bit; } ... vps_sub_picture_dividing_flag; if(vps_sub_picture_dividing_flag) { vps_full_pic_width_in_luma_samples; vps_full_pic_height_in_luma_samples; } ... }	Descriptor: u(4) u(8) v u(7) u(1) v v u(1) v ue(v) ue(v) v v v
---	--

FIG. 16A

seq_parameter_set_id; { sps_decoding_parameter_set_id; sps_video_parameter_set_id; sps_max_sub_layers_minus1; ... pic_width_in_luma_samples; pic_height_in_luma_samples; if(vps_sub_picture_dividing_flag) { pic_offset_x; pic_offset_y; } ... }	Descriptor: u(4) u(4) u(3) v ue(v) ue(v) v ue(v) ue(v) v v v
--	--

FIG. 16B

Fig. 17

sub_region_partitioning_info(payloadSize) { num_sub_region num_layers for(i = 0; i <= num_layers; i++) layer_id [i] for(i = 1; i < num_layers; i++) for(j = 0; j < i; i++) direct_dependency_flag [i][j] for(i = 0; i < num_sub_region; i++) { num_layers_for_region [i] for(j = 0; j < num_layers_for_region [i]; i++) sub_region_layer_id [i][j] sub_region_offset_x [i] sub_region_offset_y [i] sub_region_width [i] sub_region_height [i] } ... }	Descriptor
	u8(v) ²
	u8(v) ²
	i ³
	u8(v) ²
	i ³
	i ³
	u8(v) ²
	i ³
	u8(v) ²
	u8(v) ²
	u8(v) ²
	u8(v) ²
	u8(v) ²
	i ³
	i ³
	i ³

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][i++])	
output_layer_flag[i][j]	u(1)
profile_tier_level_idx[i][j]	u(v)
}	
...	
}	

Fig. 19

video parameter set {vps() {	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 (j = 0; j < num_output_layer_sets; j++) {	
vps_output_layers_model[j][i]	u(2)
vps_ptl_signal_flag[j][i]	u(1)
for (k = 0; k < NumLayersInList[j][i]; k++) {	
if (vps_output_layers_model[j][i] == 2) {	
output_layer_flag[k][i]	u(1)
if (vps_ptl_signal_flag[j][i]) {	
profile_tier_level_idx[k][i]	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[i][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 < NumLayersInDtlis[i][i][i]) {	
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][k]	u(1)
if(vps_ptl_signal_flag[i])	
profile_tier_level_idx[i][j][k]	u(v)
}	
}	
...	
}	

FIG. 21 2100

