

FIG. 1

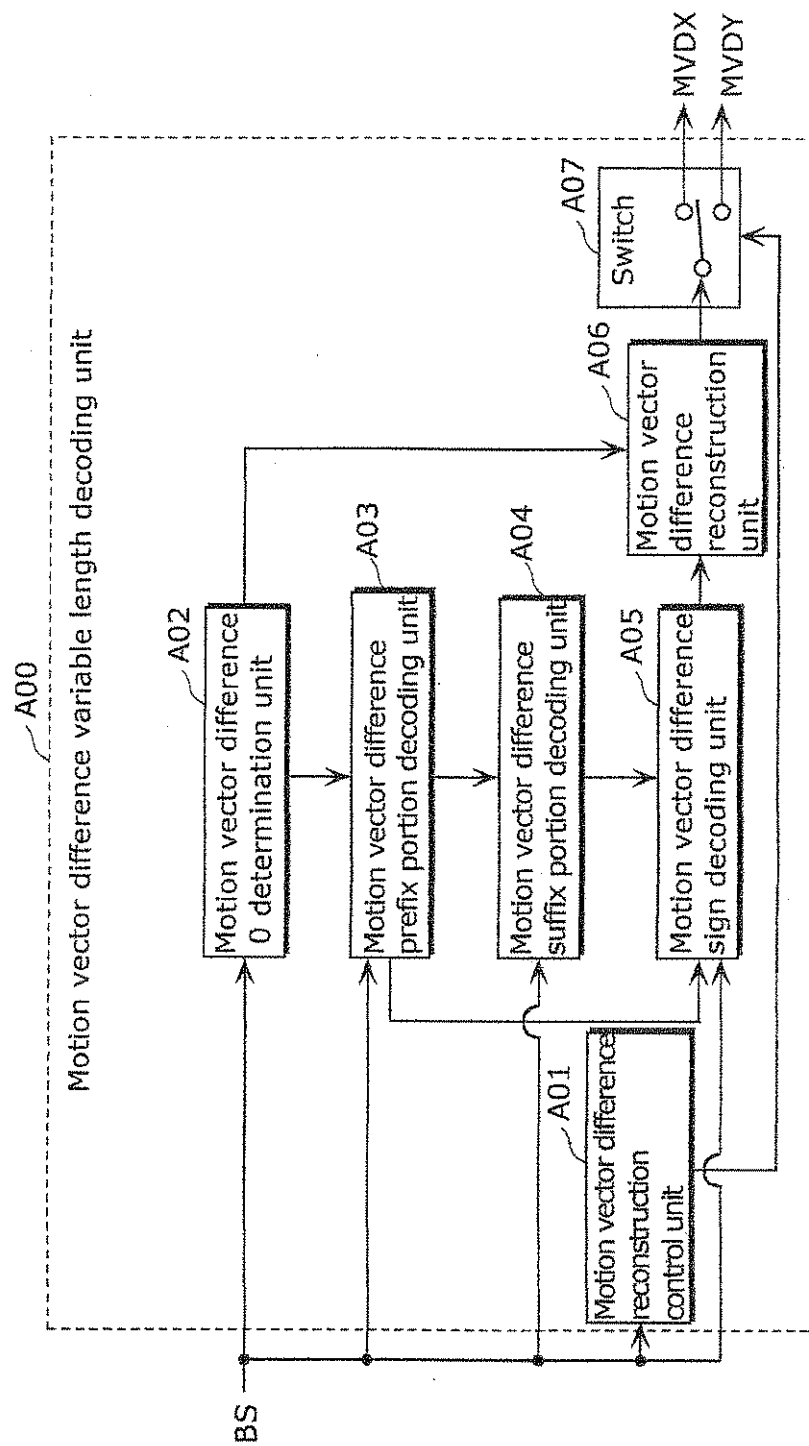


FIG. 2

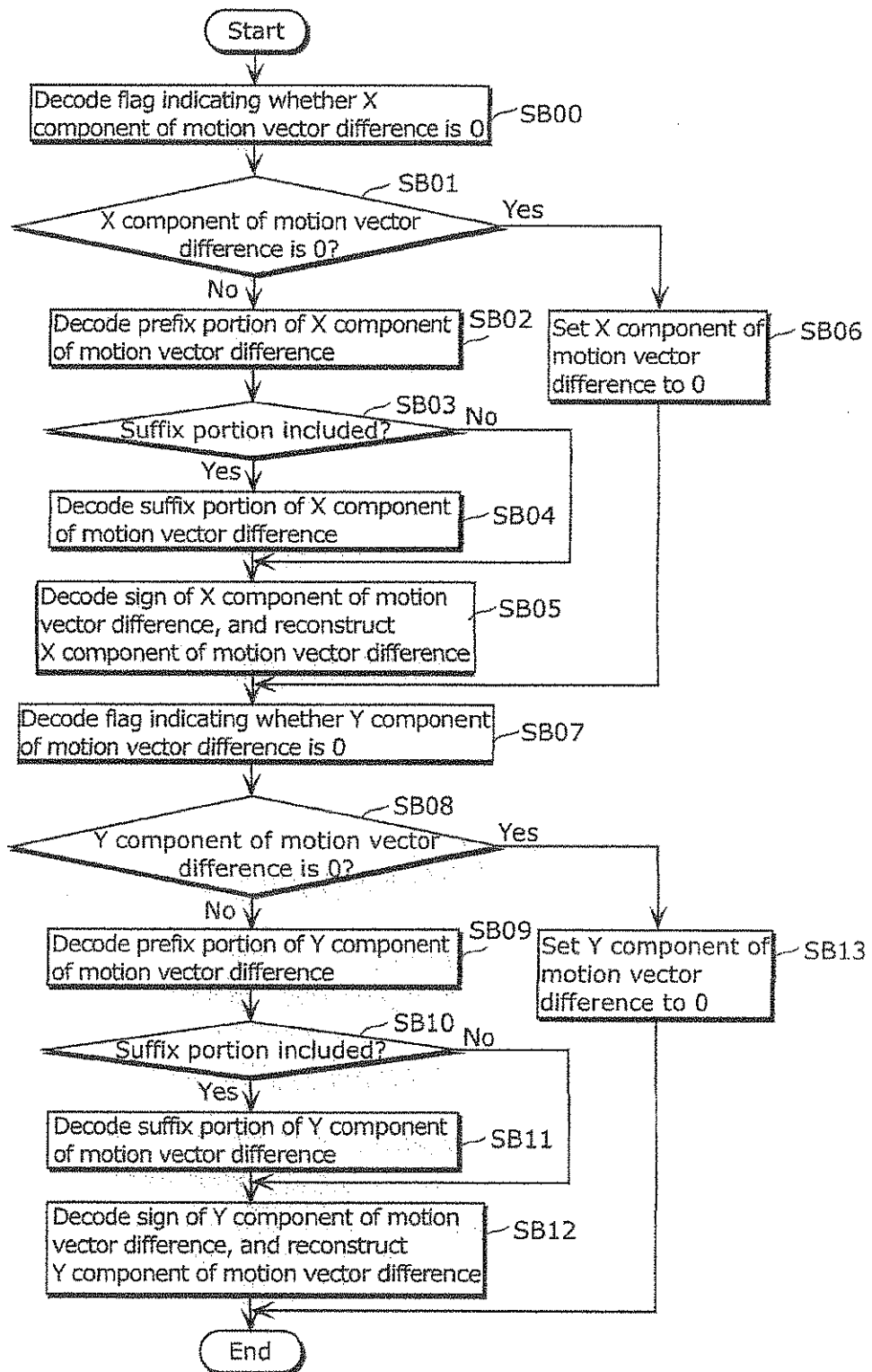


FIG. 3

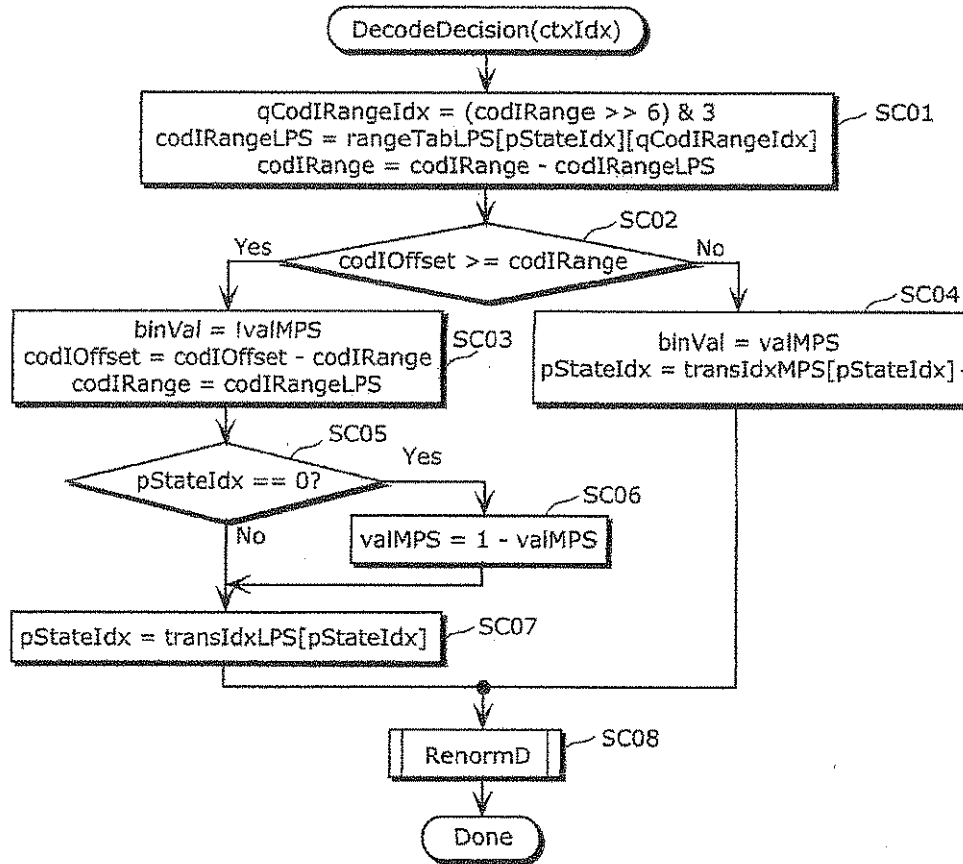


FIG. 4

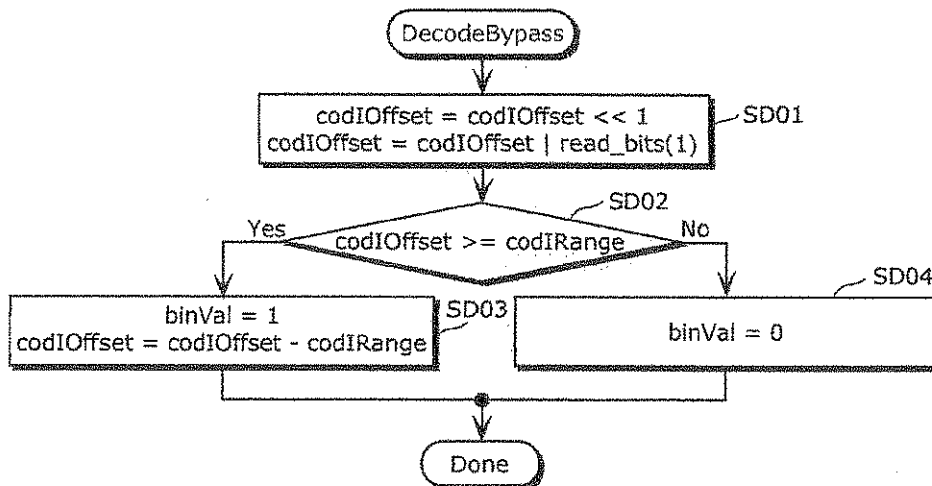


FIG. 5

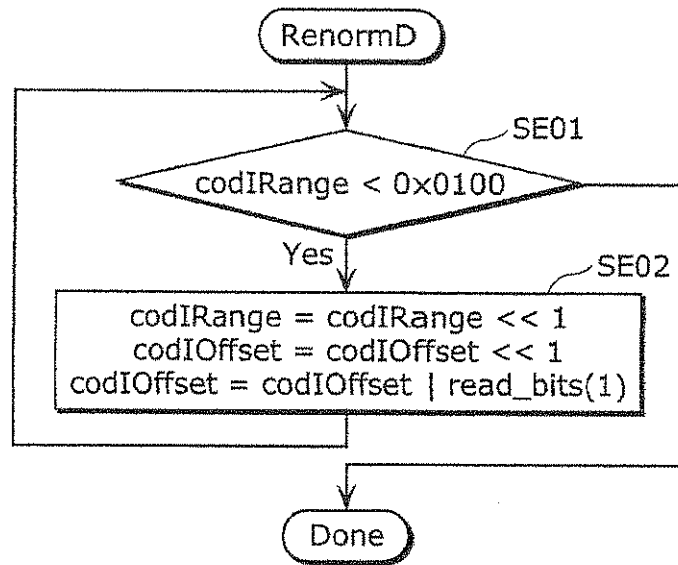


FIG. 6

Motion vector difference value	Prefix portion	Suffix portion
0	0	
1	1 0	
2	1 1 0	
3	1 1 1 0	
4	1 1 1 1 0	
5	1 1 1 1 1 0	
6	1 1 1 1 1 1 0	
7	1 1 1 1 1 1 1 0	
8	1 1 1 1 1 1 1 1 0	
9	1 1 1 1 1 1 1 1 1	0000
10	1 1 1 1 1 1 1 1 1	0001
...	...	...

FIG. 7

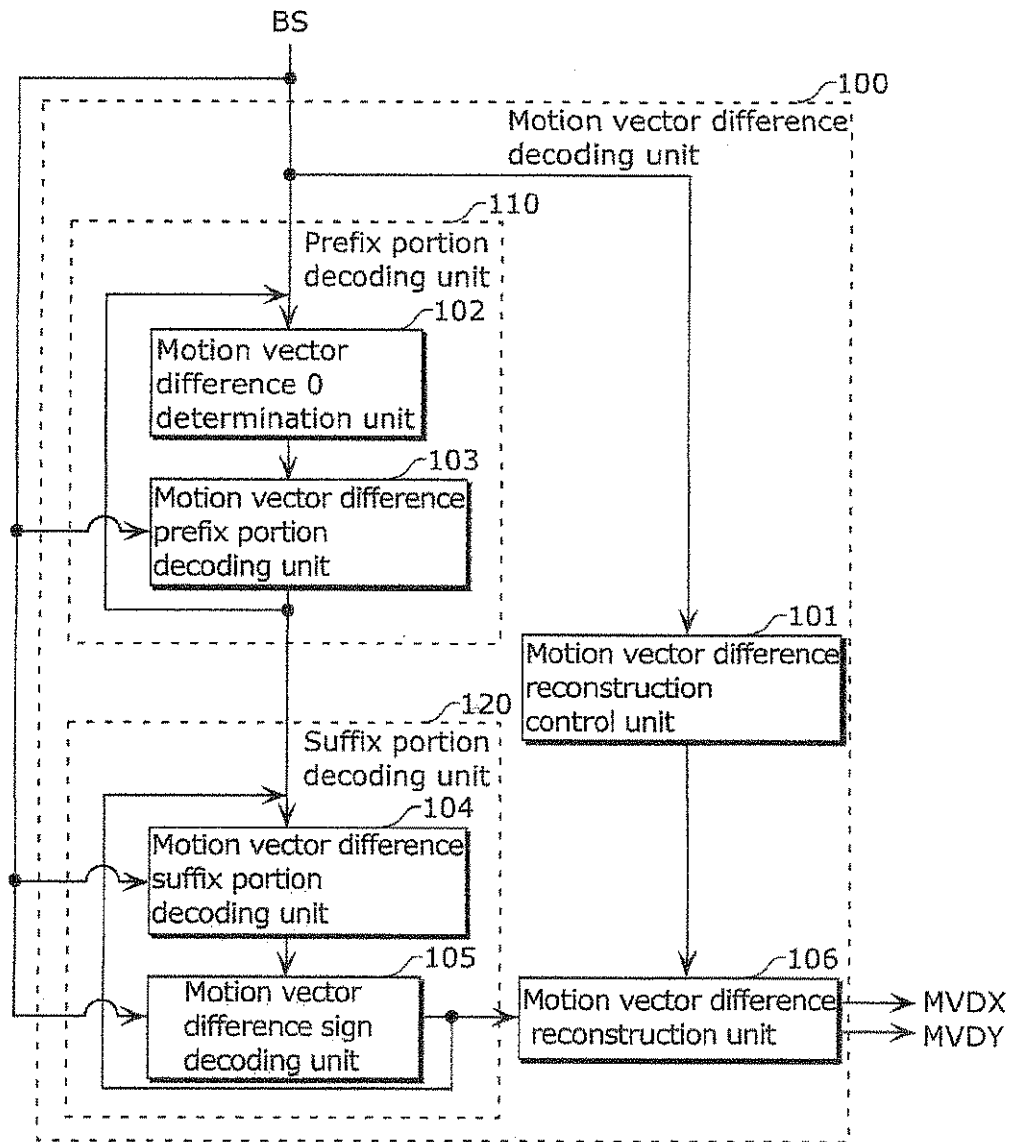


FIG. 8

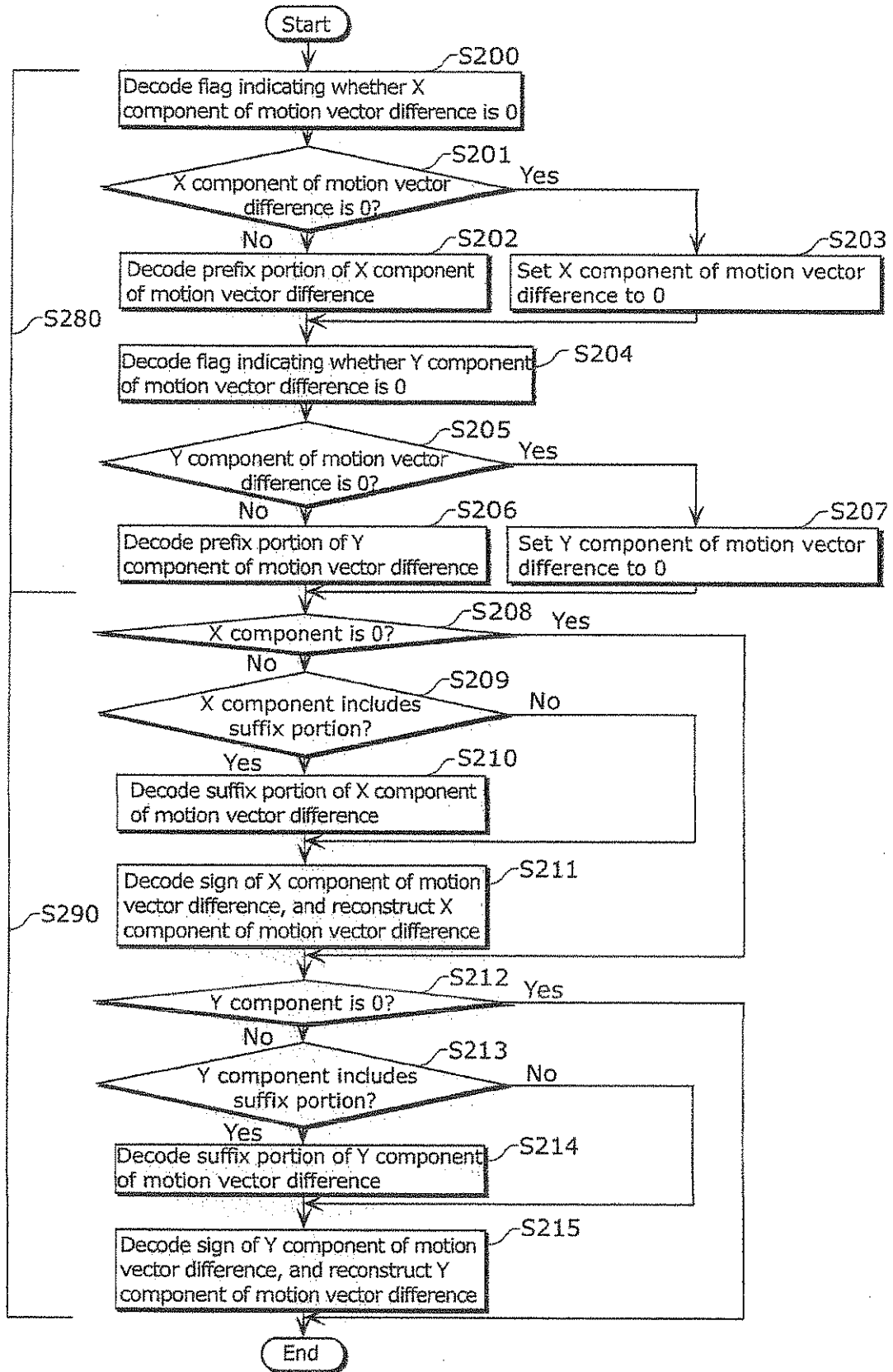


FIG. 9

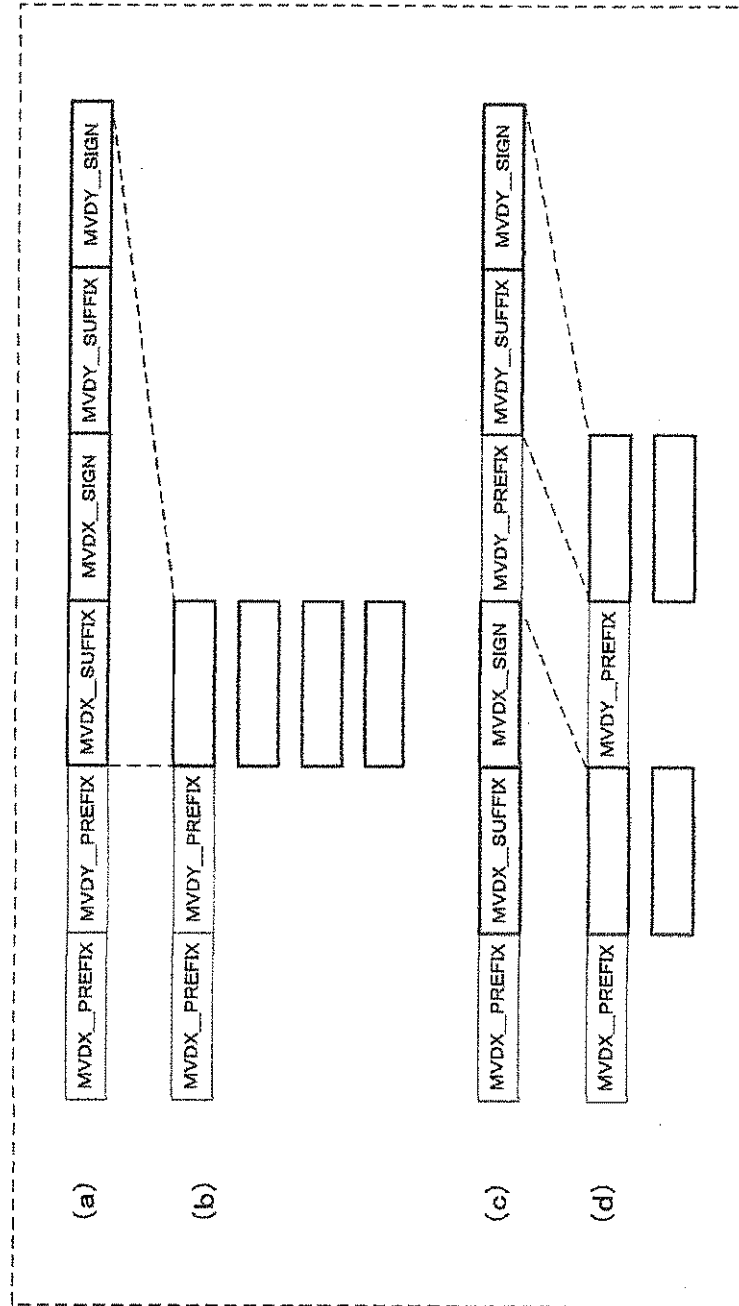


FIG. 10

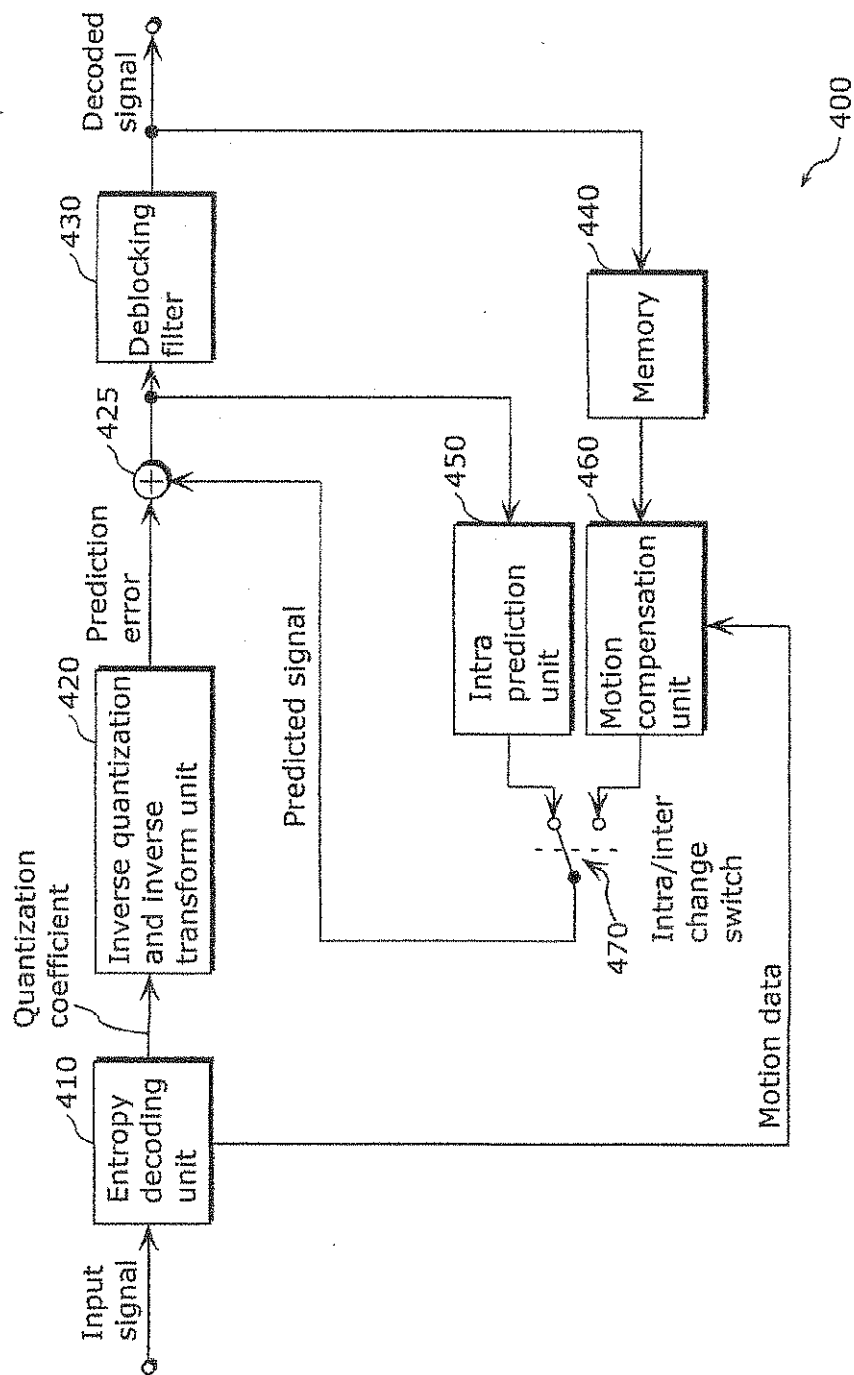


FIG. 11A

MVDXY_EXIST	0	10	110	111
MVDX_EXIST	0	1	0	1
MVDY_EXIST	0	0	1	1

FIG. 11B

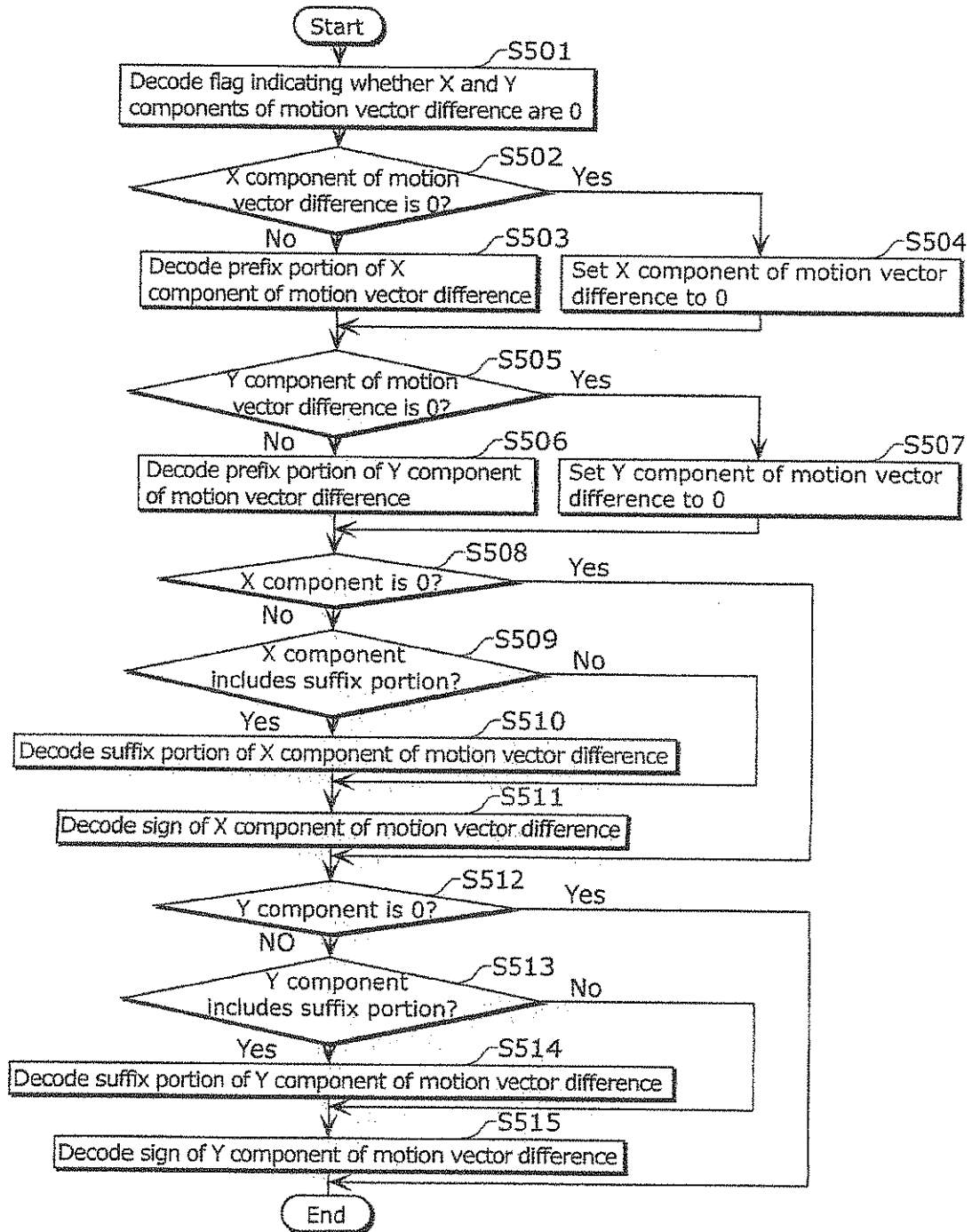


FIG. 12

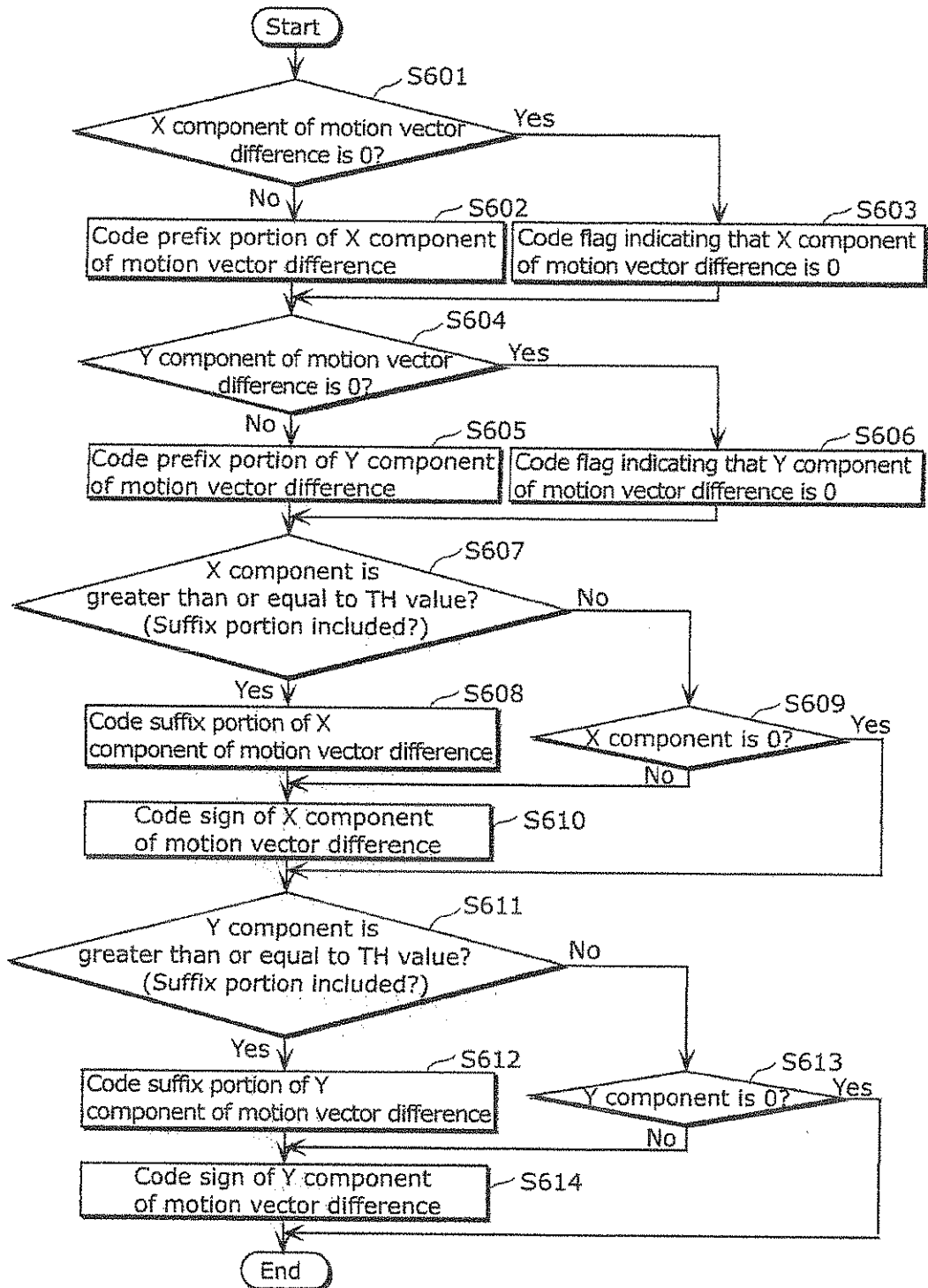


FIG. 13

		Descriptor
	prediction_unit(x0, y0, log2PUWidth, log2PUHeight, PartIdx, InferredMergeFlag) {	
	if(skip_flag[x0][y0]) {	
	if(NumMergeCand > 1)	
	merge_idx[x0][y0]	ue(v) ae(v)
	} else if(PredMode == MODE_INTRA) {	
	prev_intra_luma_pred_flag[x0][y0]	u(1) ae(v)
	if(prev_intra_luma_pred_flag[x0][y0])	
	if(NumMPMCand > 1)	
	mpm_idx[x0][y0]	u(1) ae(v)
	else	
	rem_intra_luma_pred_mode[x0][y0]	ce(v) ae(v)
	intra_chroma_pred_mode[x0][y0]	ue(v) ae(v)
	} else { /* MODE_INTER */	
	if(!InferredMergeFlag)	
	if(entropy_coding_mode_flag PartMode != PART_2Nx2N)	
	merge_flag[x0][y0]	u(1) ae(v)
	if(merge_flag[x0][y0] && NumMergeCand > 1) {	
	merge_idx[x0][y0]	ue(v) ae(v)
	} else {	
	if(slice_type == B)	
	inter_pred_flag[x0][y0]	ue(v) ae(v)
	if(inter_pred_flag[x0][y0] == Pred_LC) {	
	if(num_ref_idx_lc_active_minus1 > 0)	
	ref_idx_lc[x0][y0]	ue(v) ae(v)
701	mvd_lc[x0][y0]	se(v) ae(v)
	if(NumMVPCand(LcToLx) > 1)	
	mvp_idx_lc[x0][y0]	ue(v) ae(v)
	}	
	else { /* Pred_L0 or Pred_BI */	
	if(num_ref_idx_l0_active_minus1 > 0)	
	ref_idx_l0[x0][y0]	ue(v) ae(v)
702	mvd_l0[x0][y0]	se(v) ae(v)
	if(NumMVPCand(L0) > 1)	
	mvp_idx_l0[x0][y0]	ue(v) ae(v)
	}	
	if(inter_pred_flag[x0][y0] == Pred_BI) {	
	if(num_ref_idx_l1_active_minus1 > 0)	
	ref_idx_l1[x0][y0]	ue(v) ae(v)
703	mvd_l1[x0][y0]	se(v) ae(v)
	if(NumMvPCand(L1) > 1)	
	mvp_idx_l1[x0][y0]	ue(v) ae(v)
	}	
	}	
	}	
	}	
	}	
	}	

FIG. 14

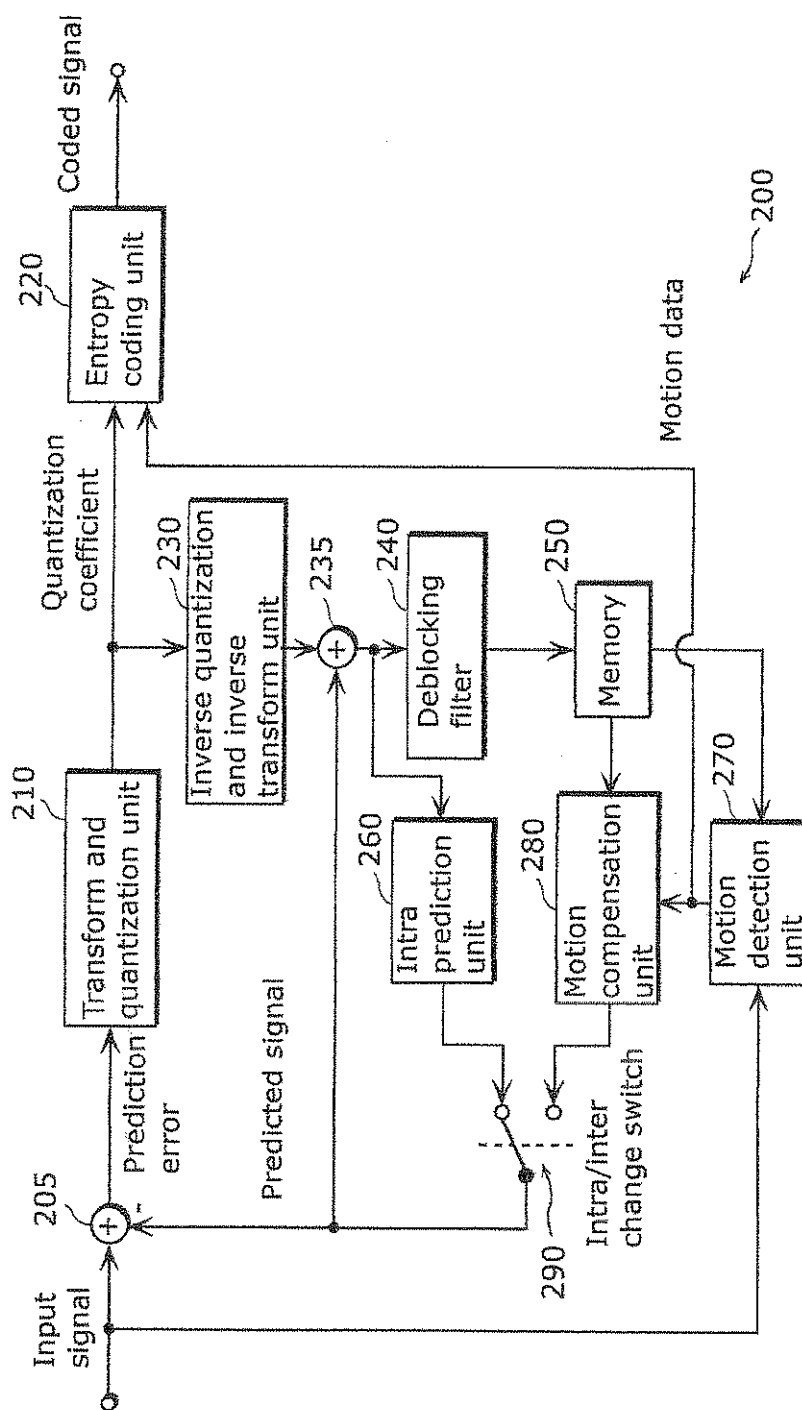


FIG. 15A

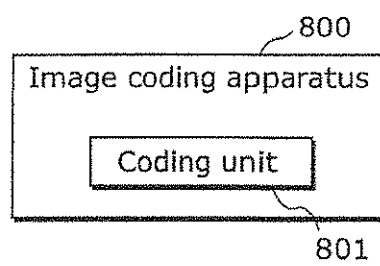


FIG. 15B

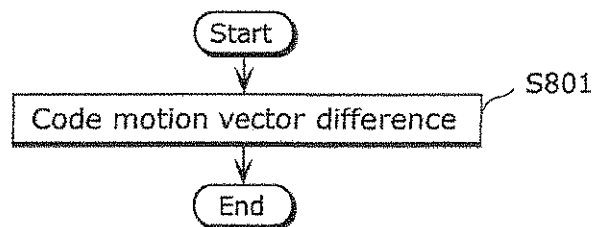


FIG. 16A

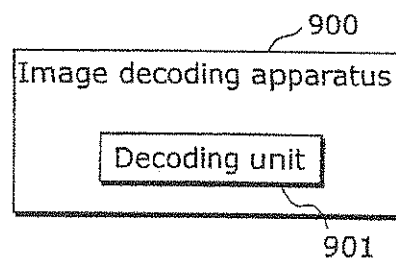


FIG. 16B

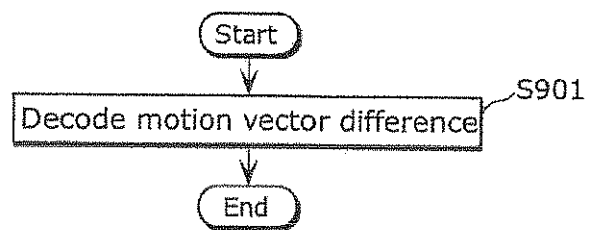


FIG. 17

mvd_coding(mvd_x, mvd_y) {		
abs_mvd_greater0_flag[0]		ae(v)
abs_mvd_greater0_flag[1]		ae(v)
if(abs_mvd_greater0_flag[0])		
abs_mvd_greater1_flag[0]		ae(v)
if(abs_mvd_greater0_flag[1])		
abs_mvd_greater1_flag[1]		ae(v)
if(abs_mvd_greater0_flag[0]) {		
if(abs_mvd_greater1_flag[0])		
abs_mvd_minus2[0]		ae(v)
mvd_sign_flag[0]		ae(v)
}		
if(abs_mvd_greater0_flag[1]) {		
if(abs_mvd_greater1_flag[1])		
abs_mvd_minus2[1]		ae(v)
mvd_sign_flag[1]		ae(v)
}		
mvd_x = abs_mvd_greater0_flag[0] * (abs_mvd_minus2[0] + 2) *		
(1 - 2 * mvd_sign_flag[0])		
mvd_y = abs_mvd_greater0_flag[1] * (abs_mvd_minus2[1] + 2) *		
(1 - 2 * mvd_sign_flag[1])		
}		

FIG. 18

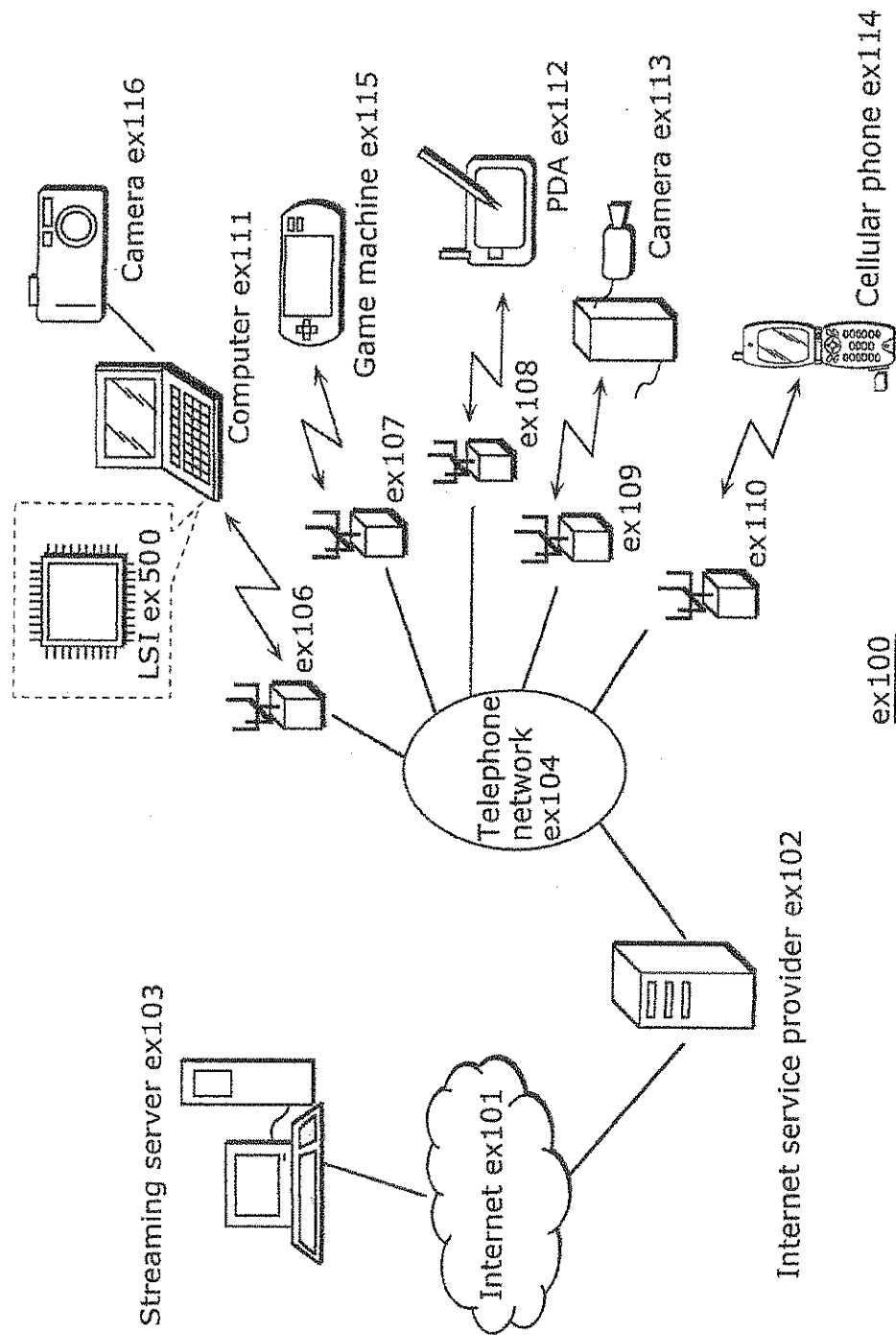


FIG. 19

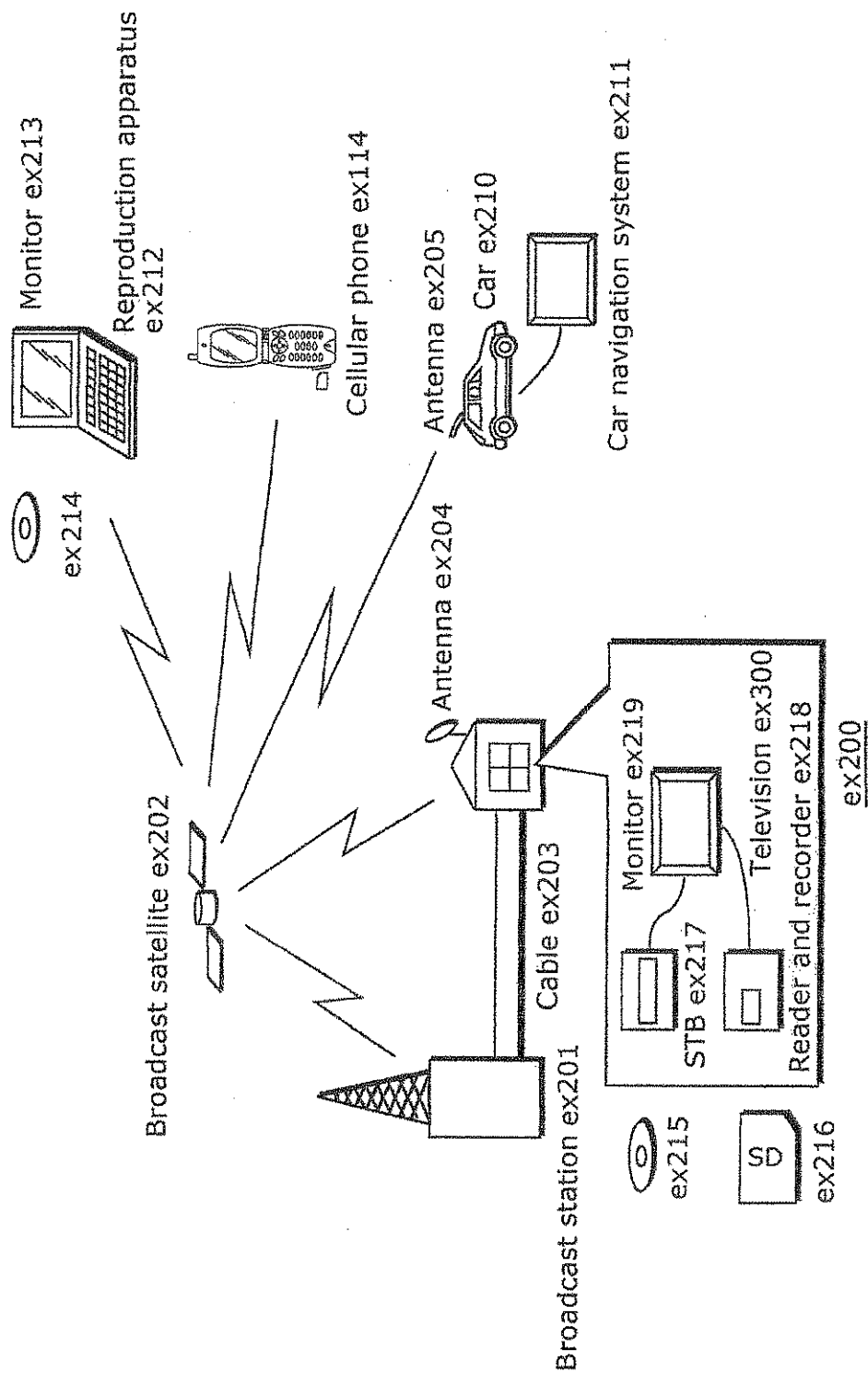


FIG. 20

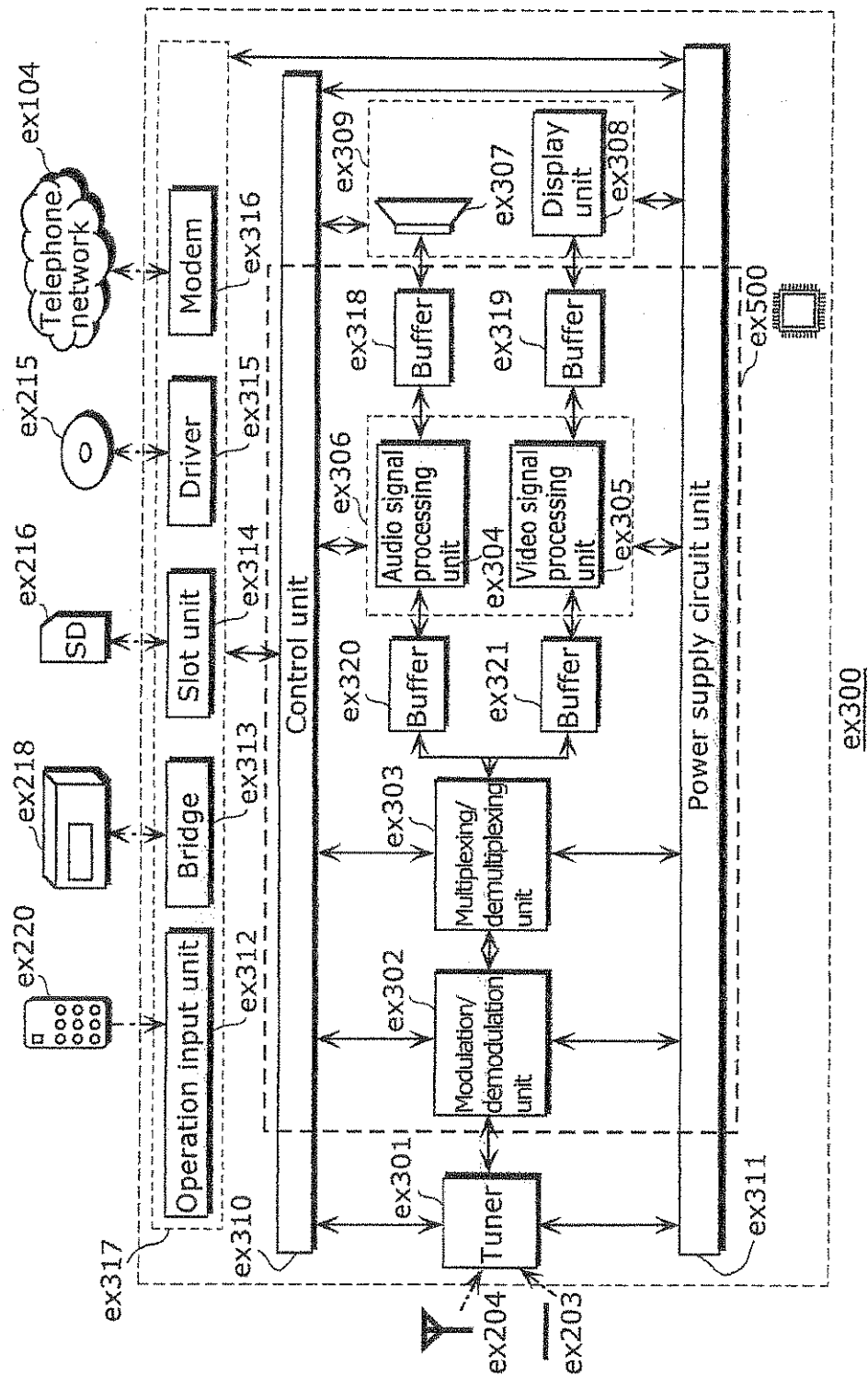


FIG. 21

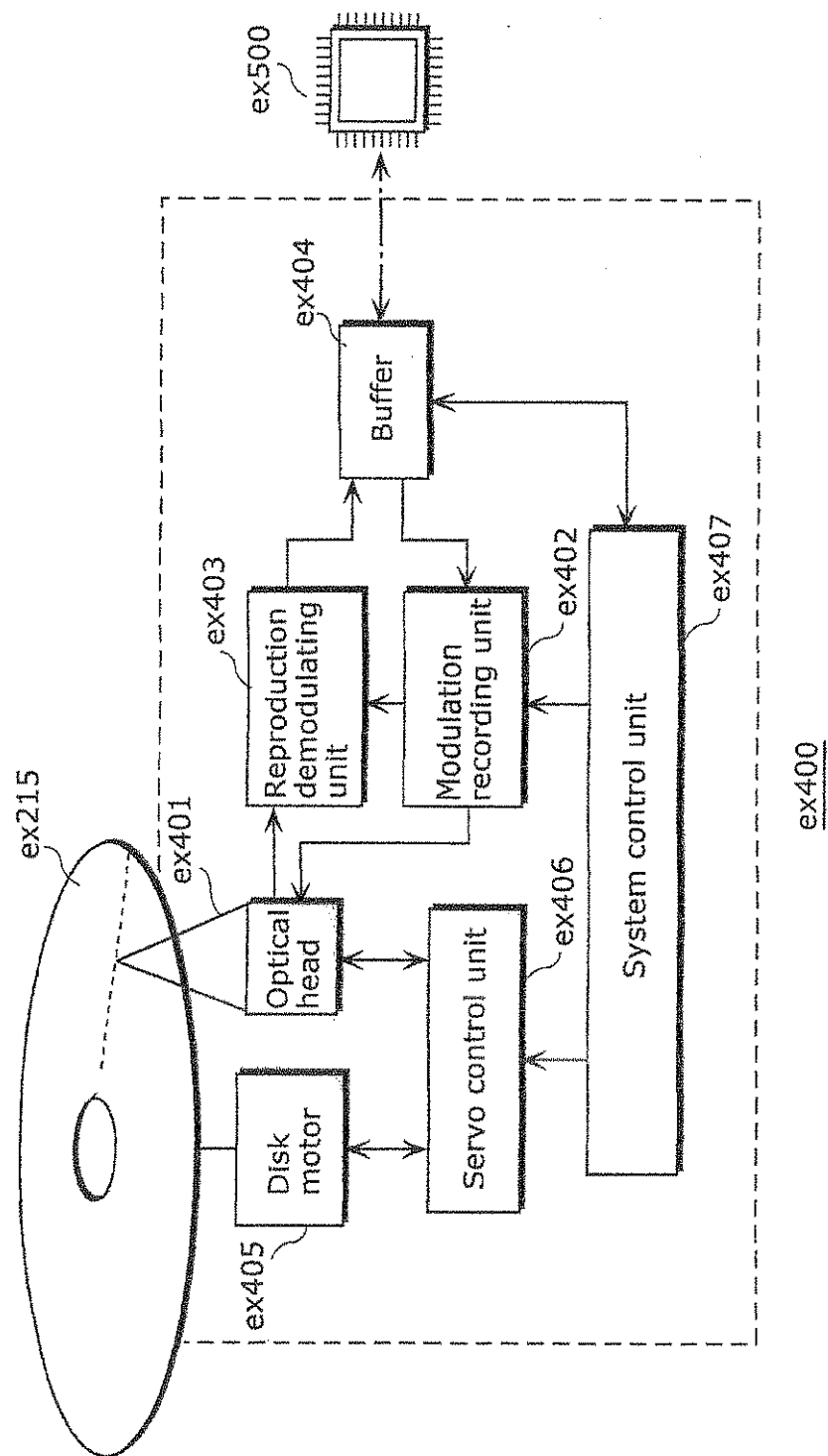


FIG. 22

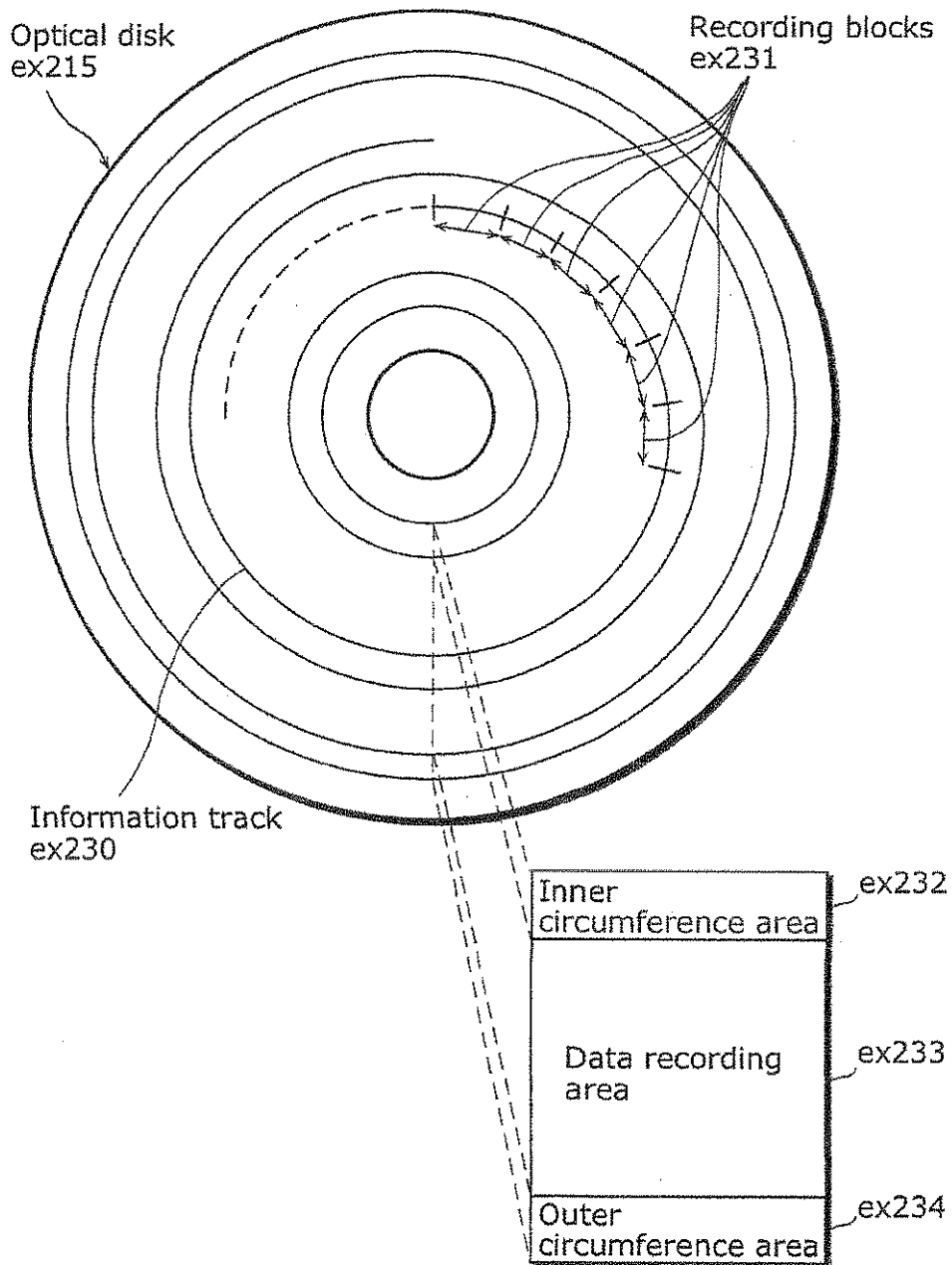


FIG. 23A

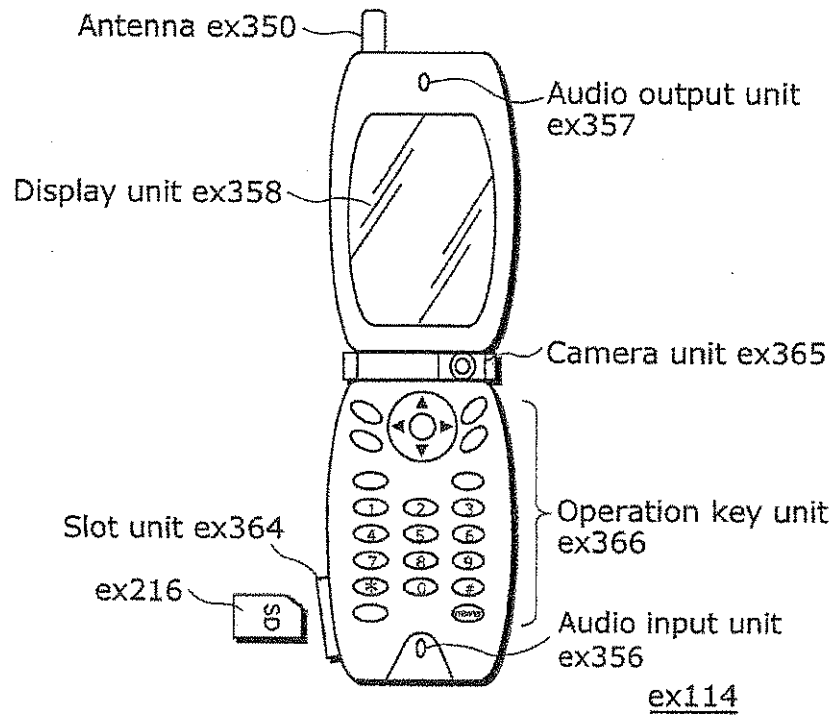


FIG. 23B

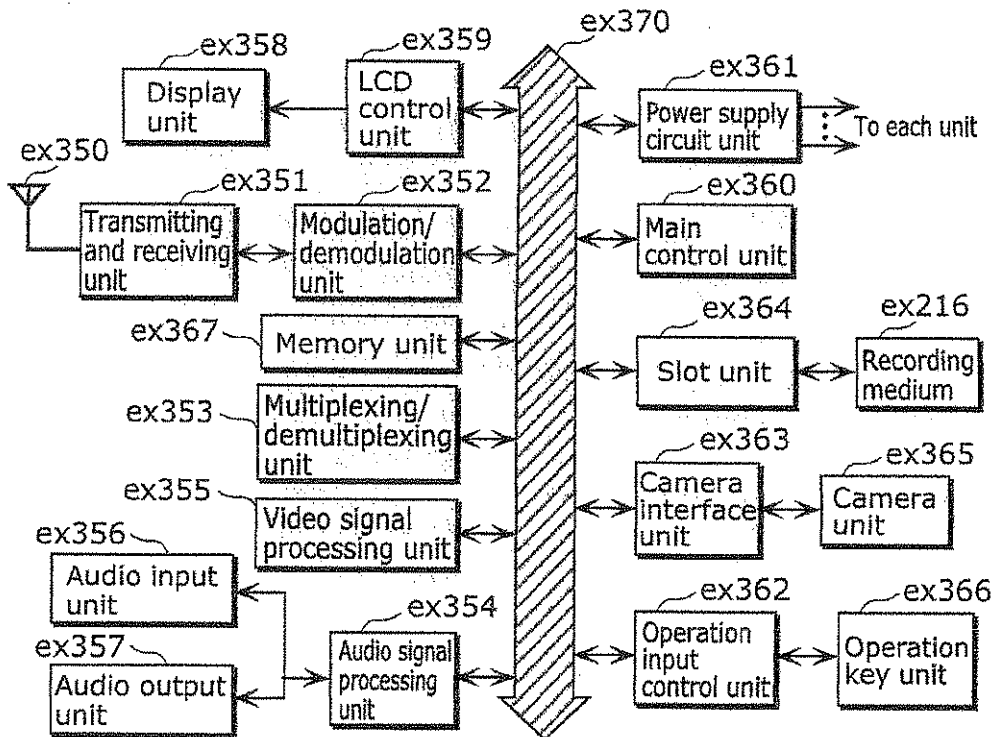


FIG. 24

Video stream (PID=0x1011, Primary video)
Audio stream (PID=0x1100)
Audio stream (PID=0x1101)
Presentation graphics stream (PID=0x1200)
Presentation graphics stream (PID=0x1201)
Interactive graphics stream (PID=0x1400)
Video stream (PID=0x1B00, Secondary video)
Video stream (PID=0x1B01, Secondary video)

FIG. 25

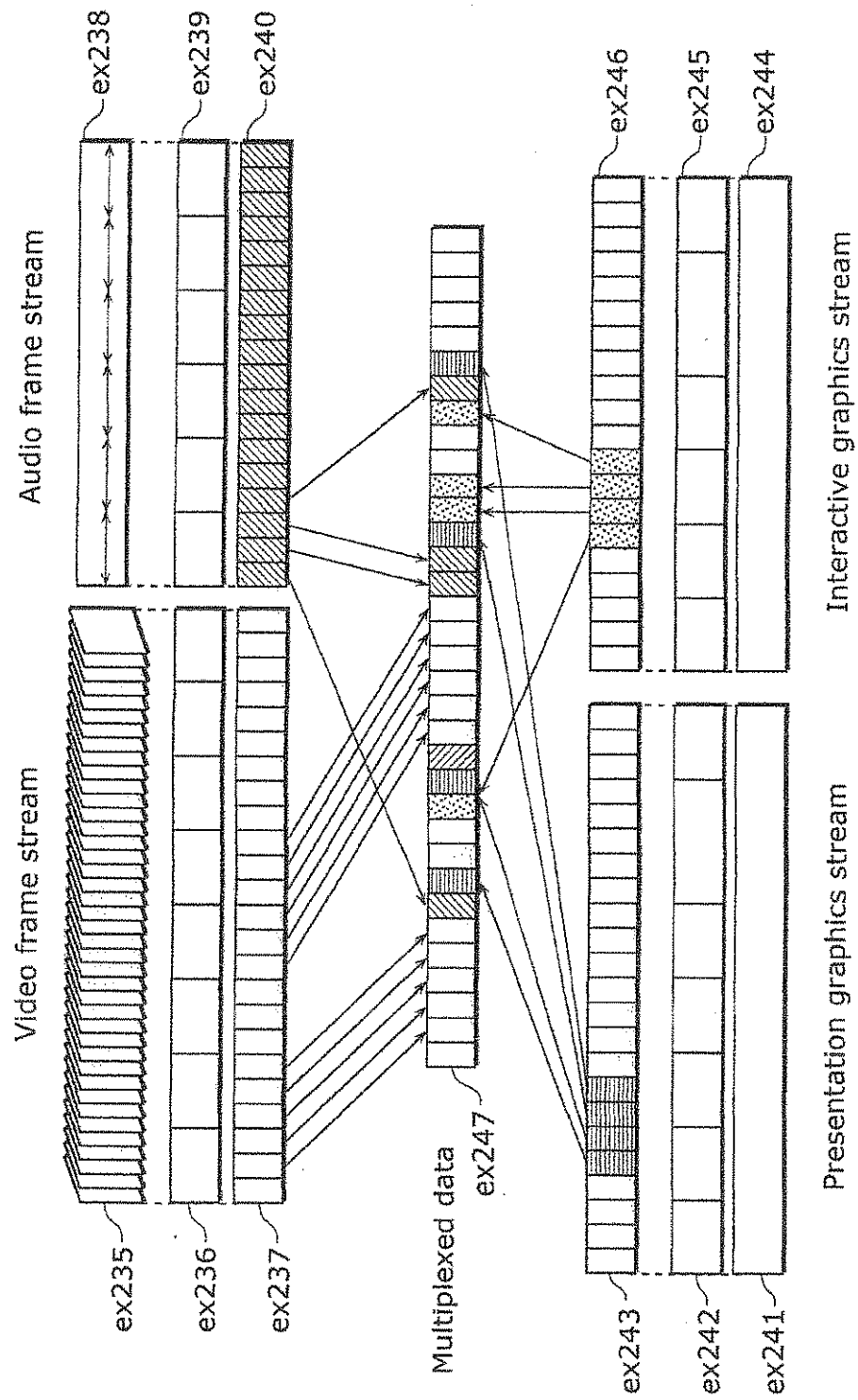


FIG. 26

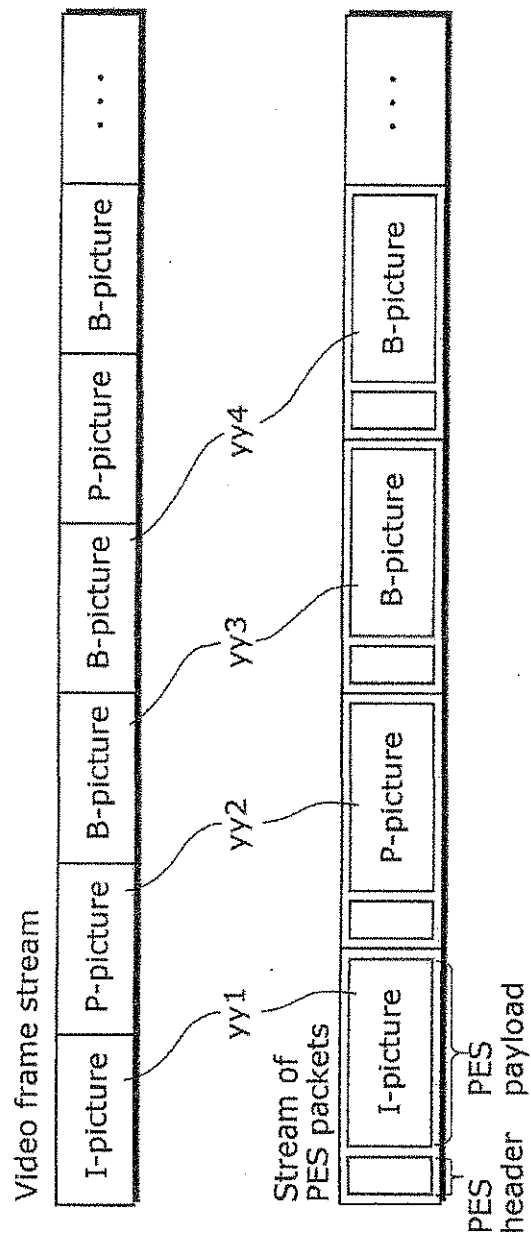
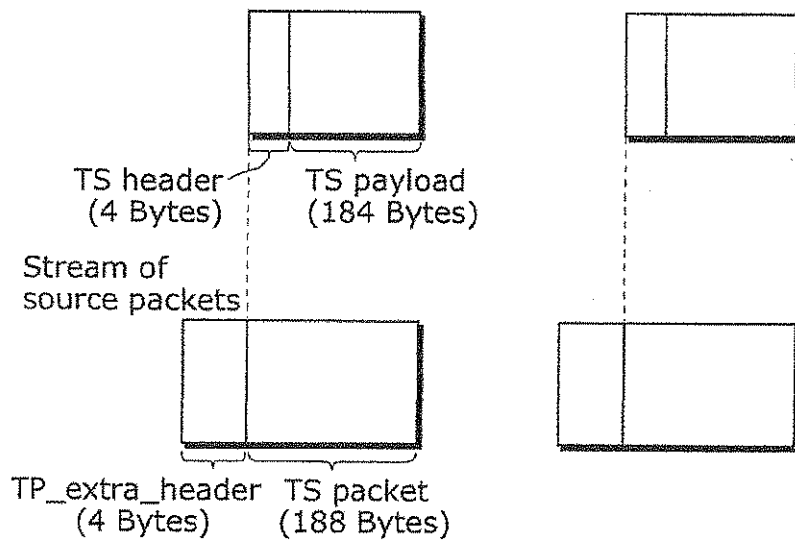


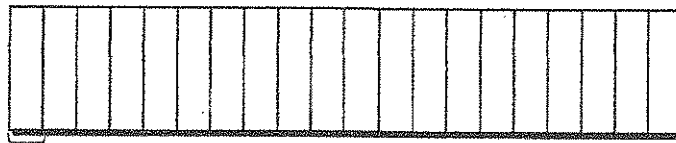
FIG. 27

Stream of TS packets



Multiplexed data

SPN 0 1 2 3 4 5 6 7 ...



Source packet

FIG. 28

Data structure of PMT

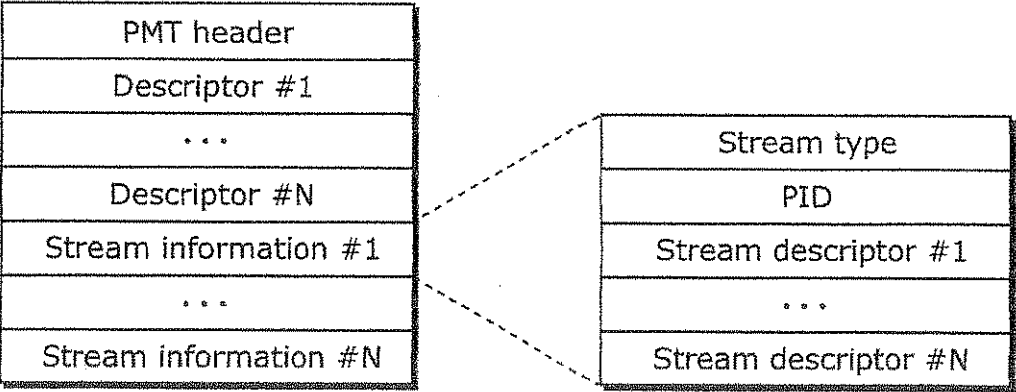


FIG. 29

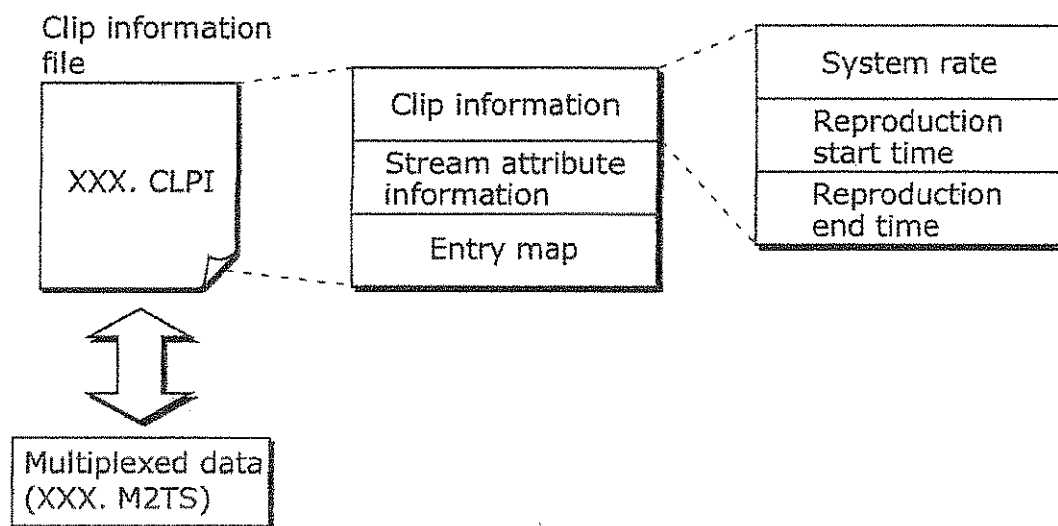


FIG. 30

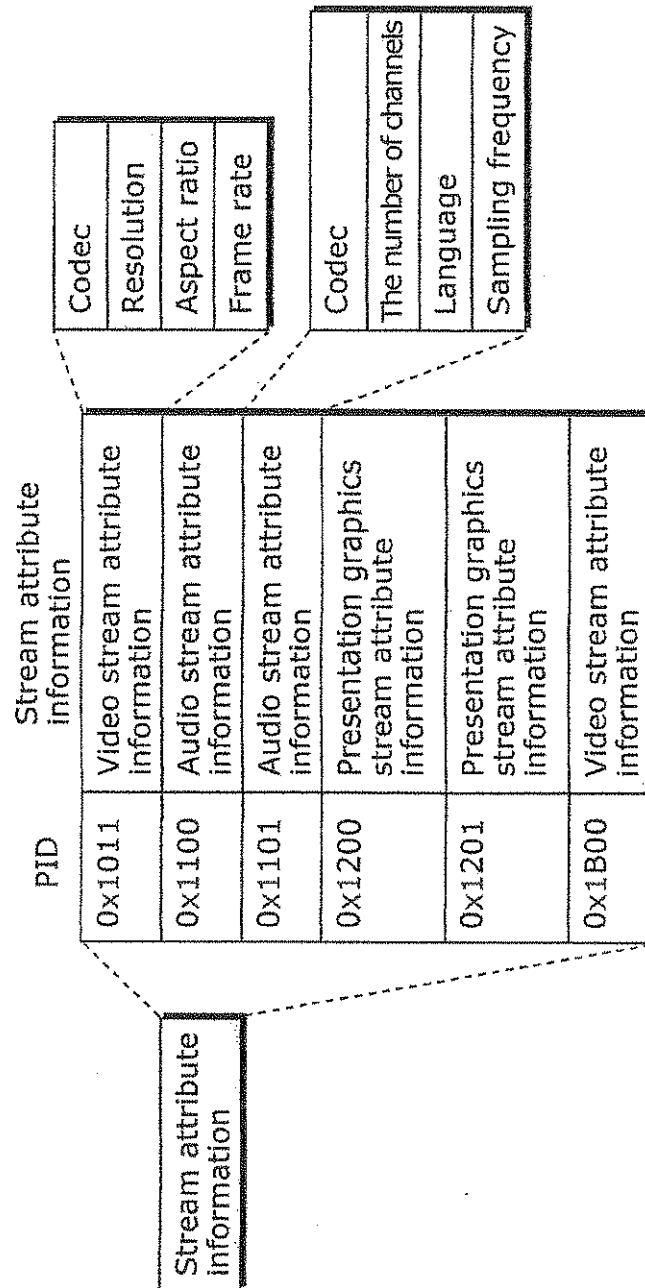


FIG. 31

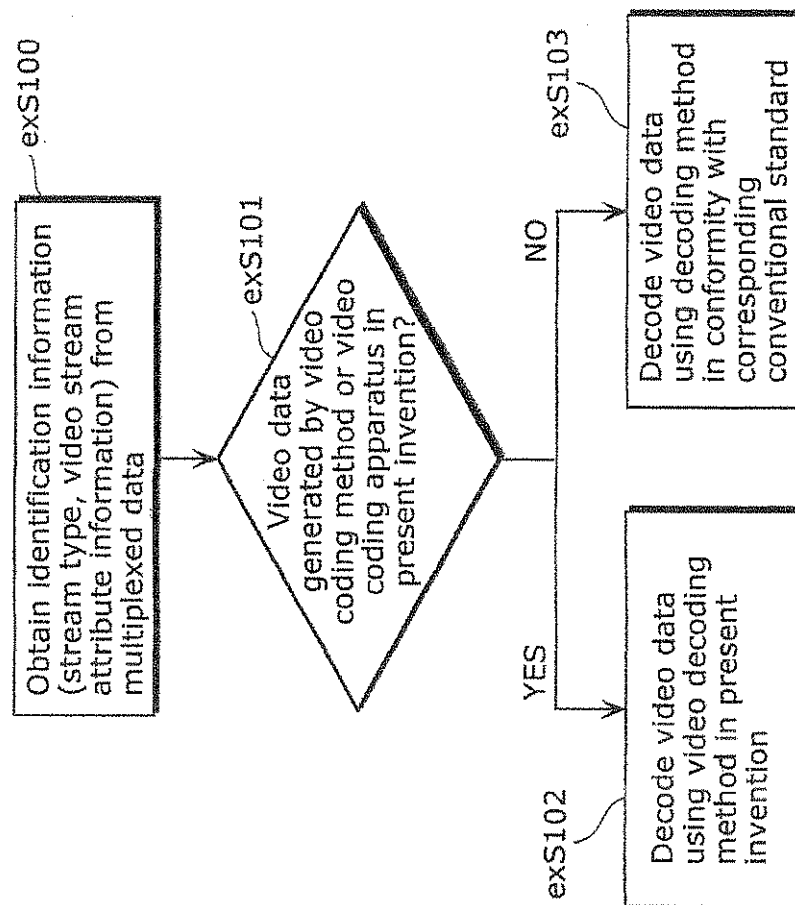


FIG. 32

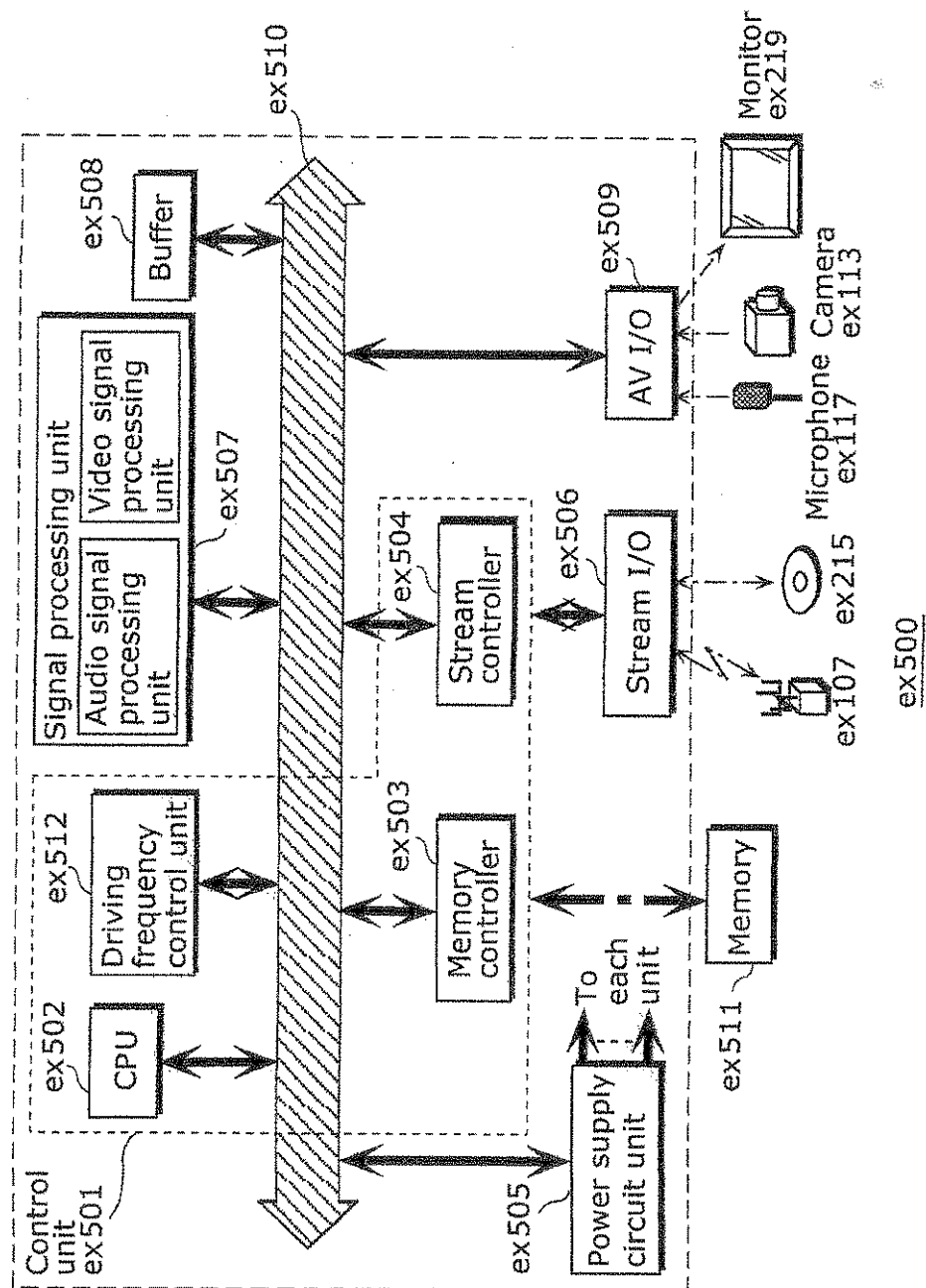


FIG. 33

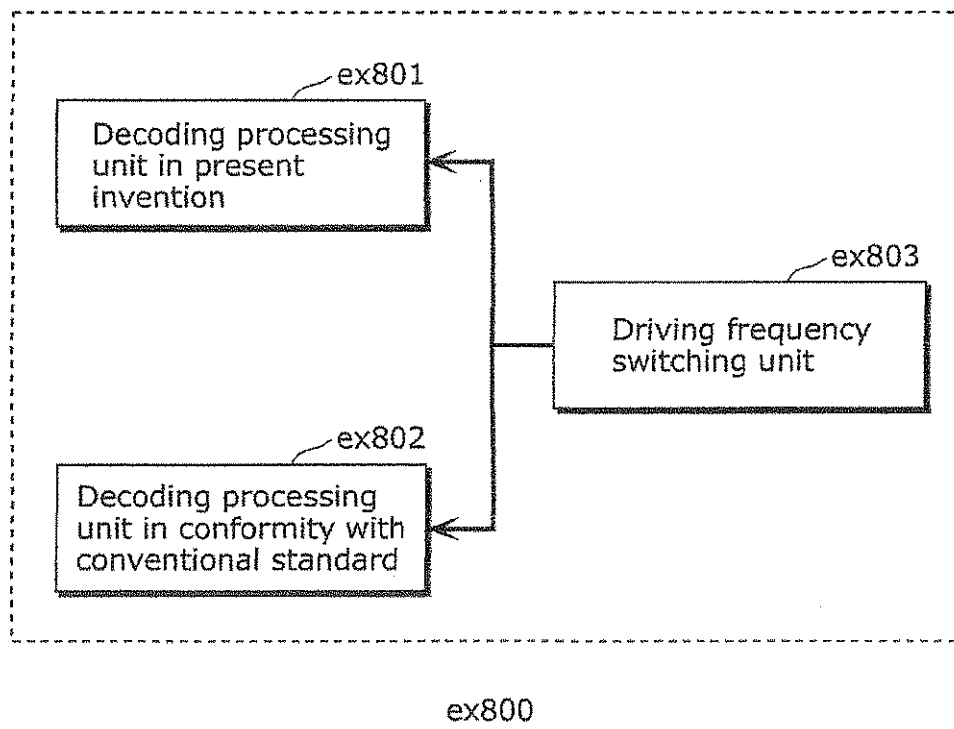


FIG. 34

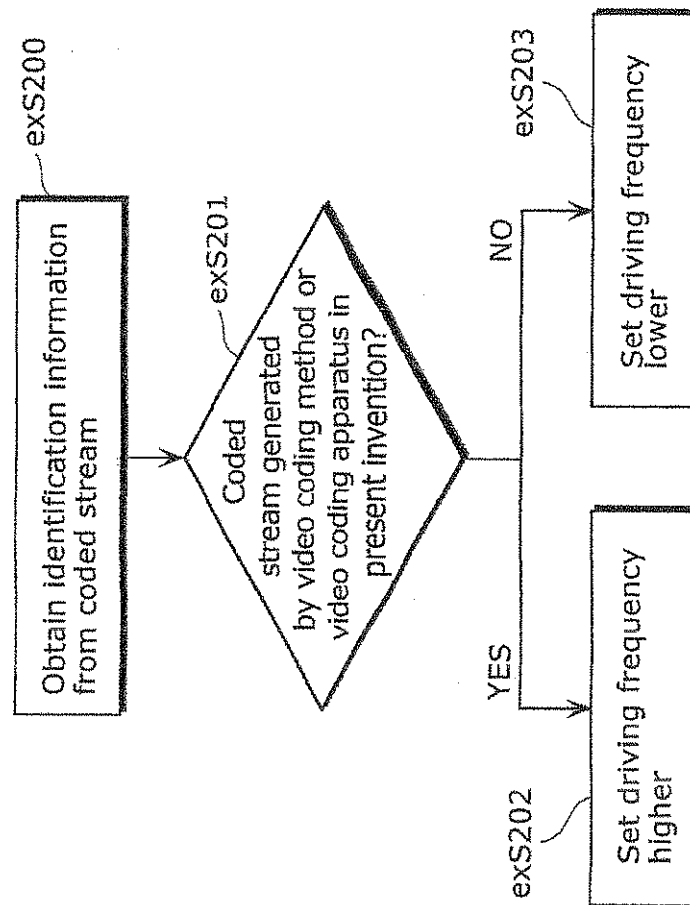


FIG. 35

Corresponding standard	Driving frequency
MPEG-4 AVC	500 MHz
MPEG-2	350 MHz
⋮	⋮

FIG. 36A

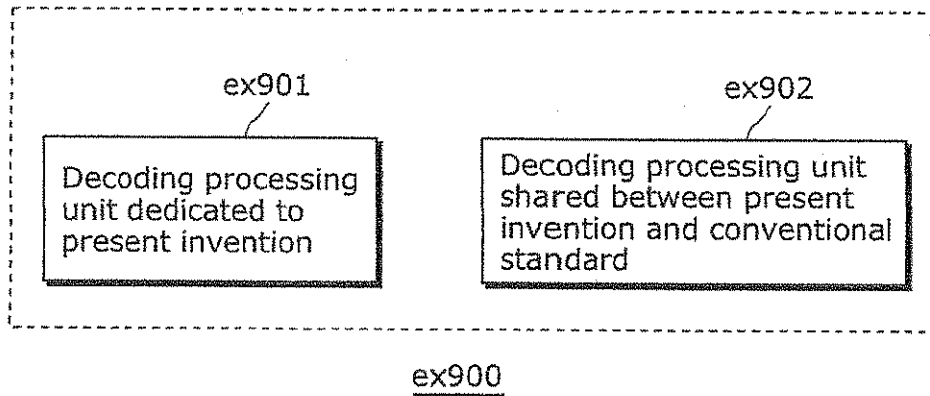


FIG. 36B

