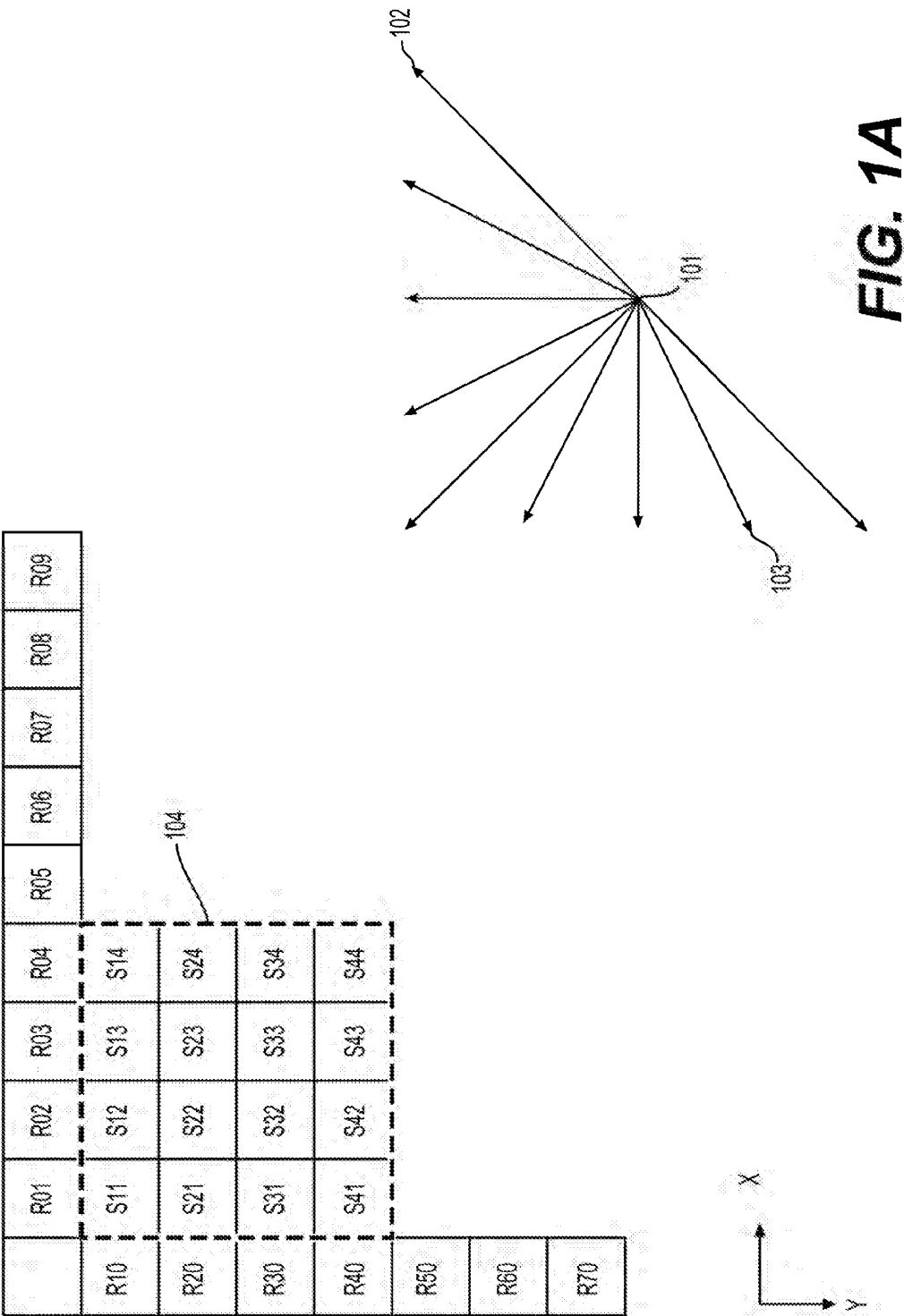




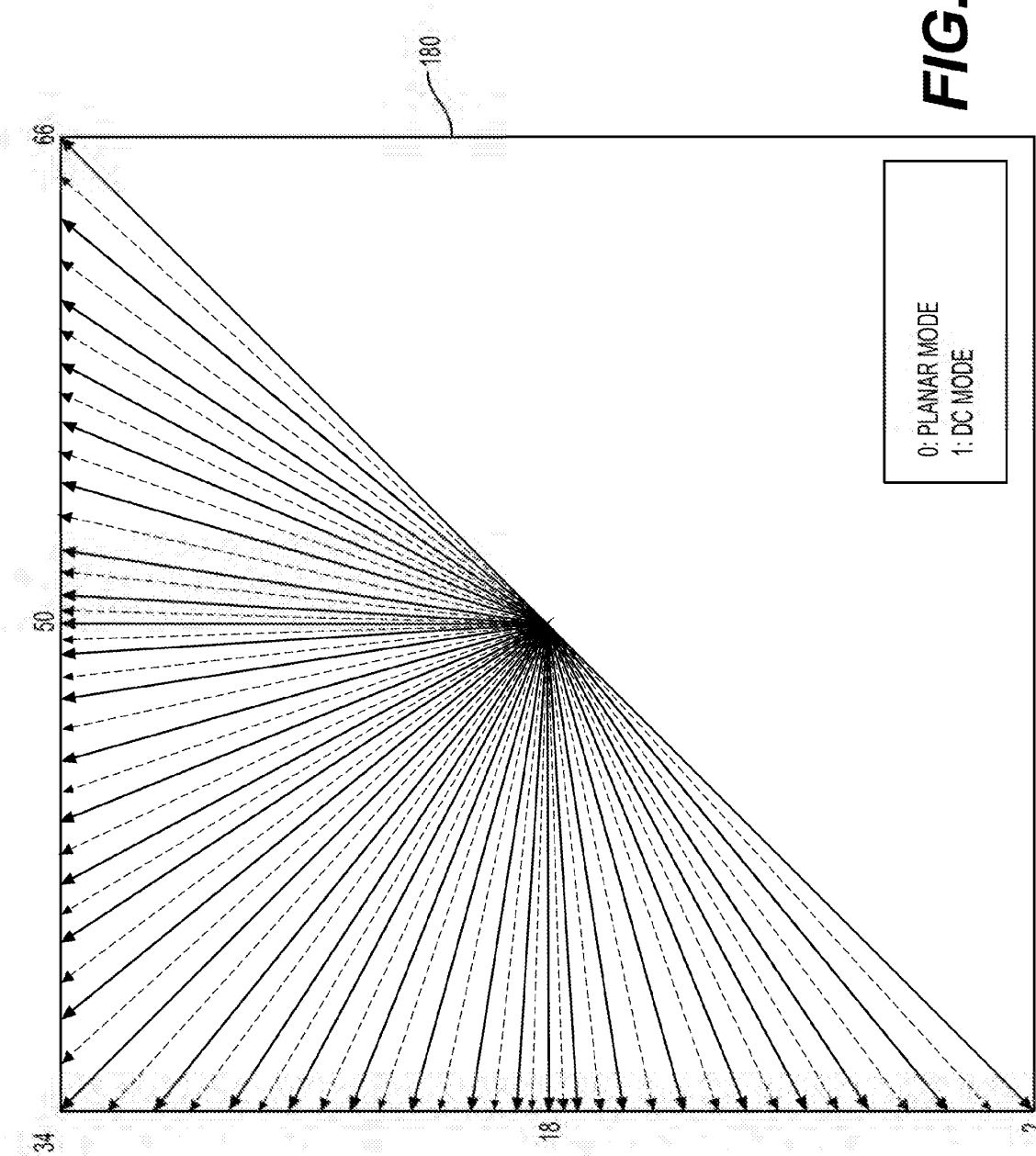


FIG. 1B

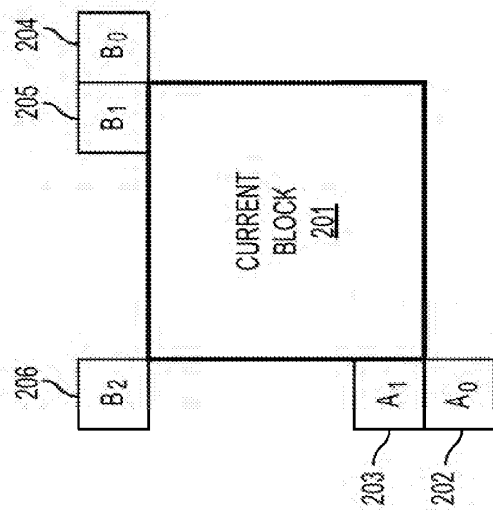


FIG. 2
(Related Art)

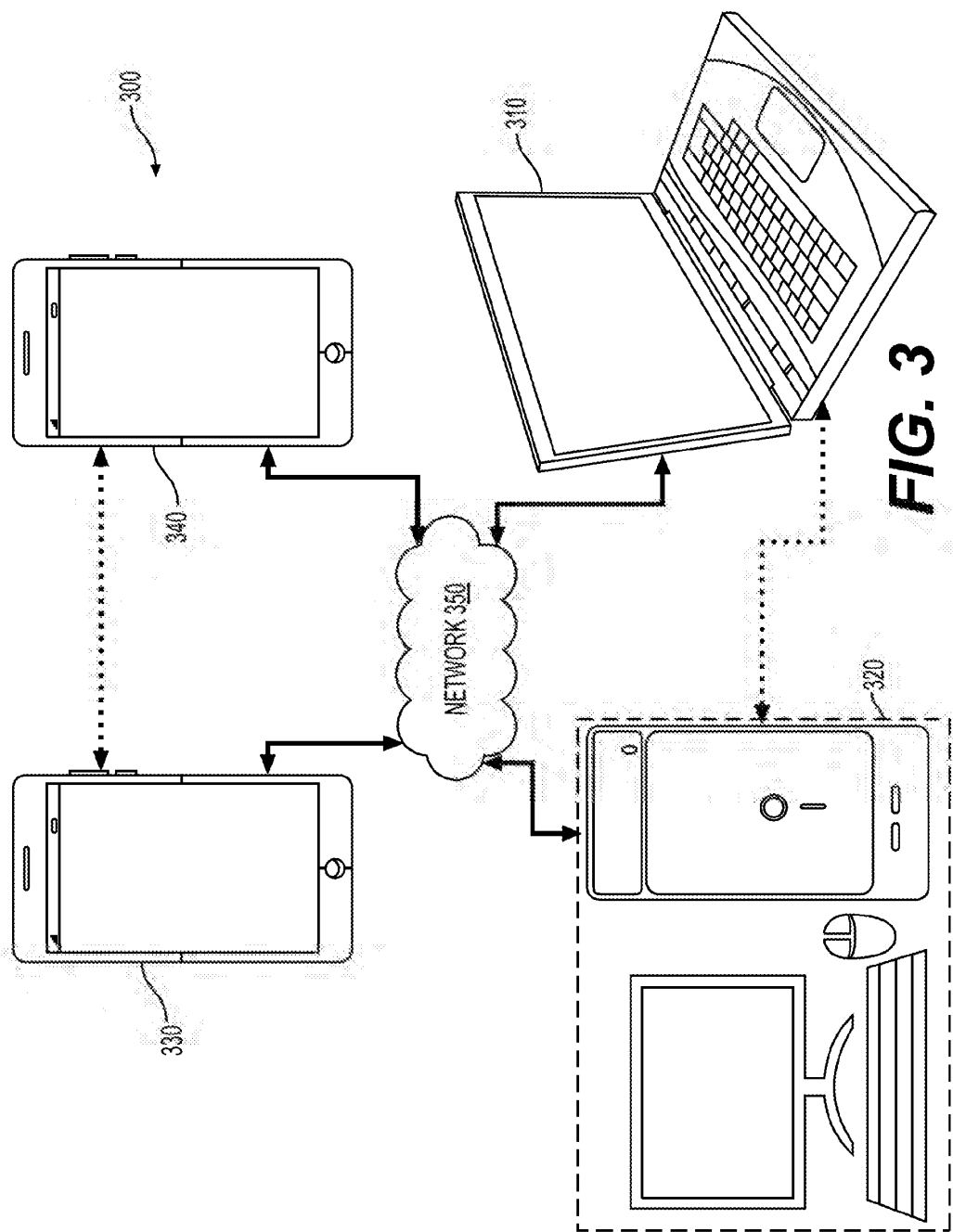


FIG. 3

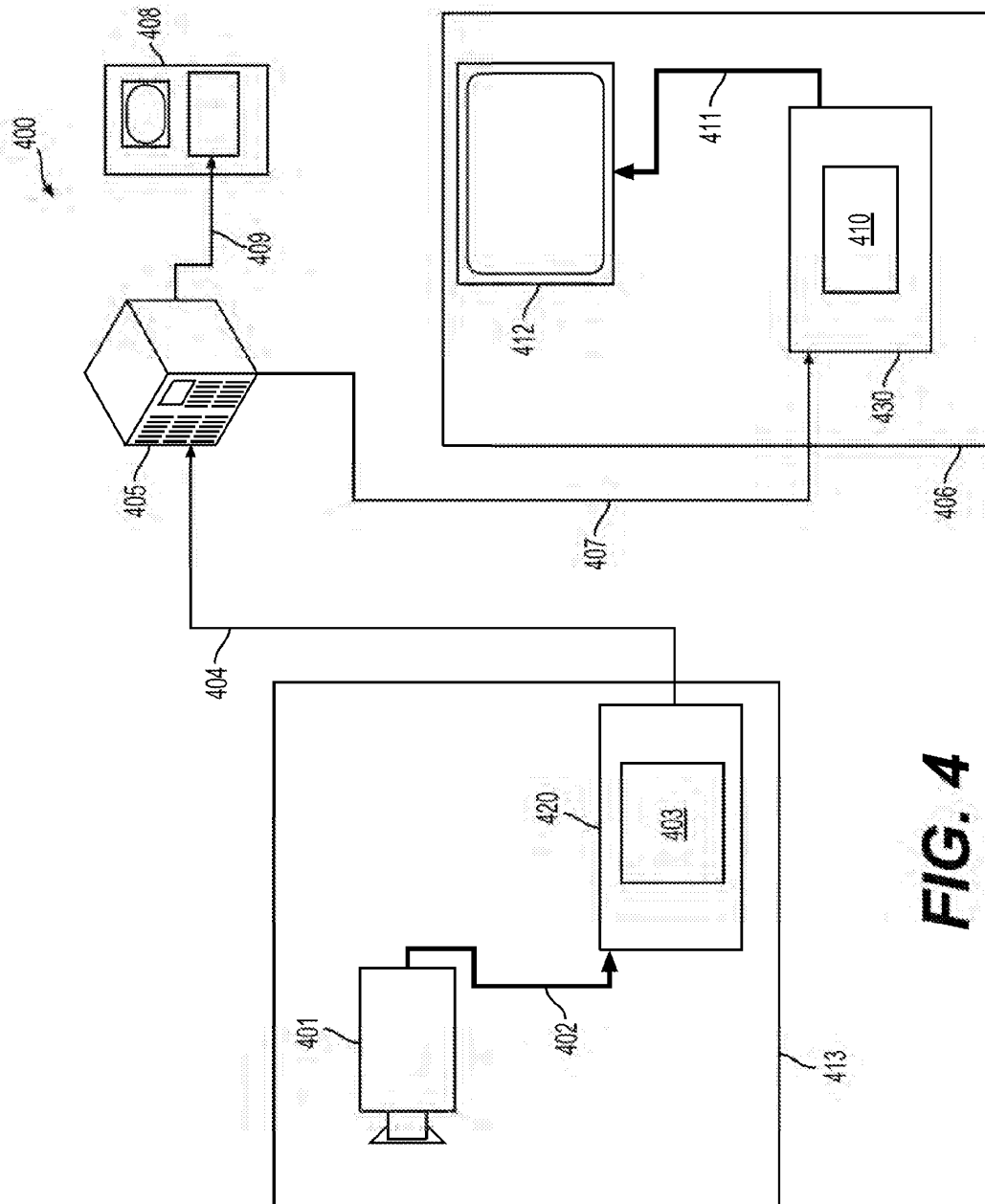


FIG. 4

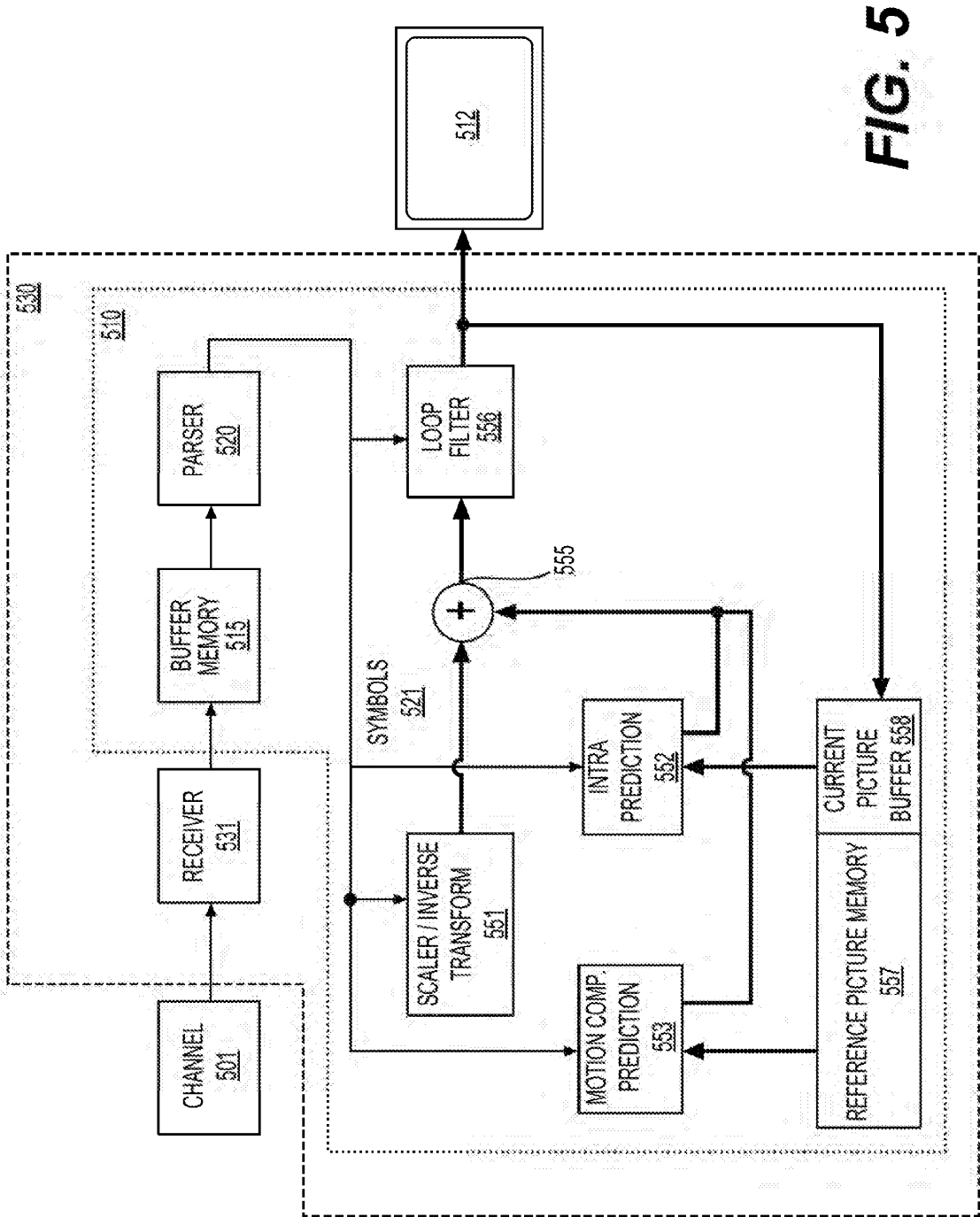


FIG. 5

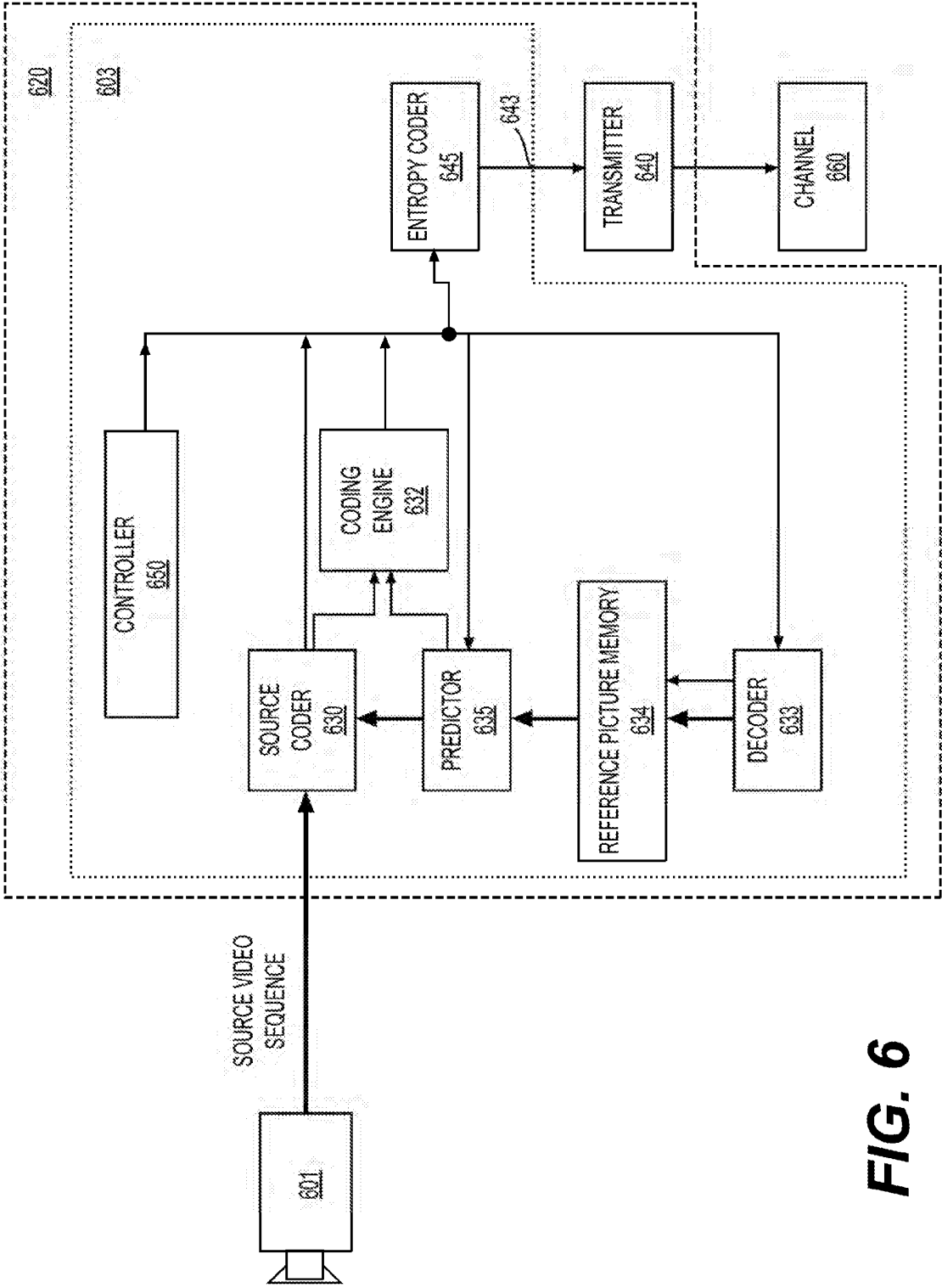


FIG. 6

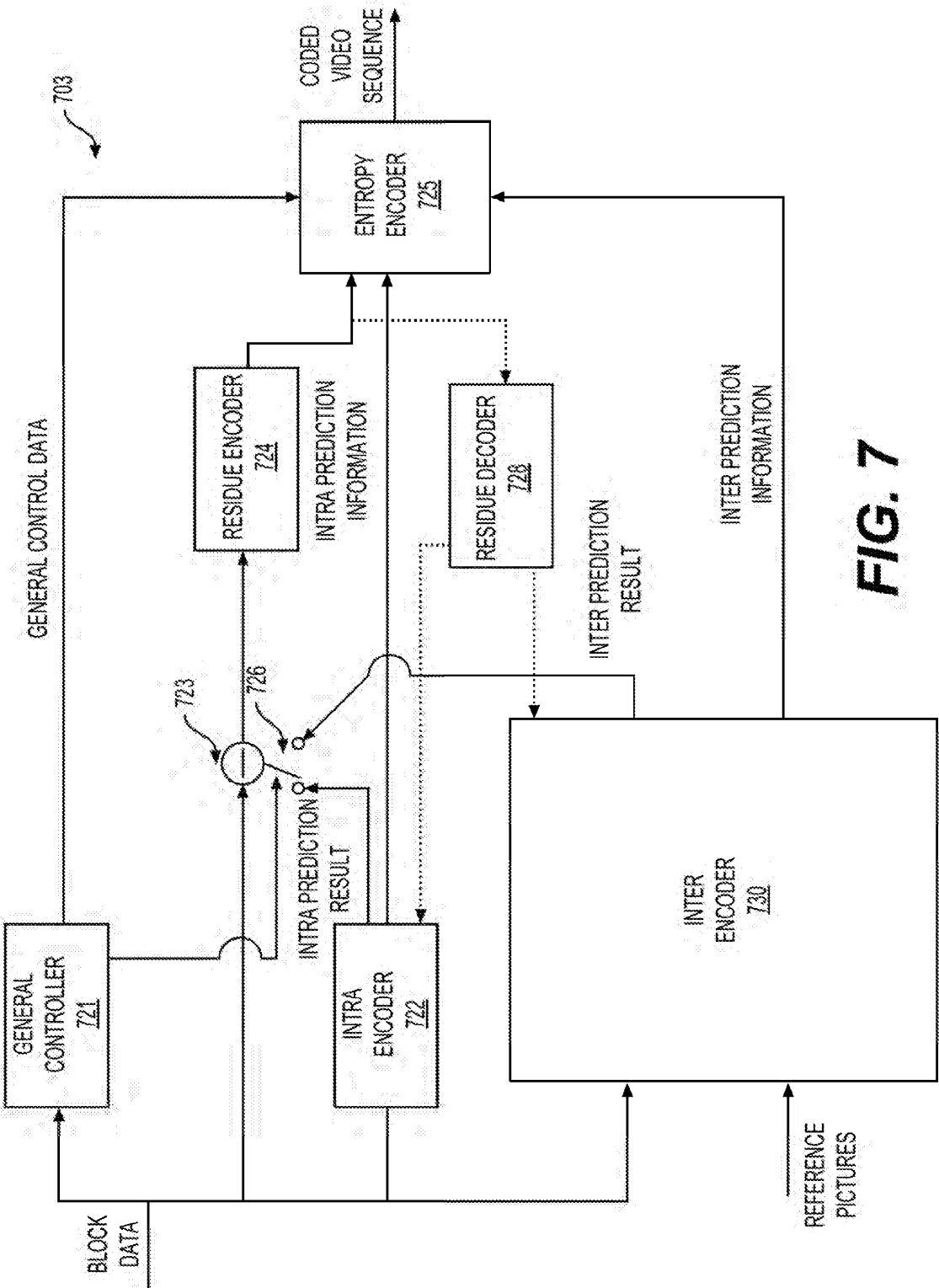


FIG. 7

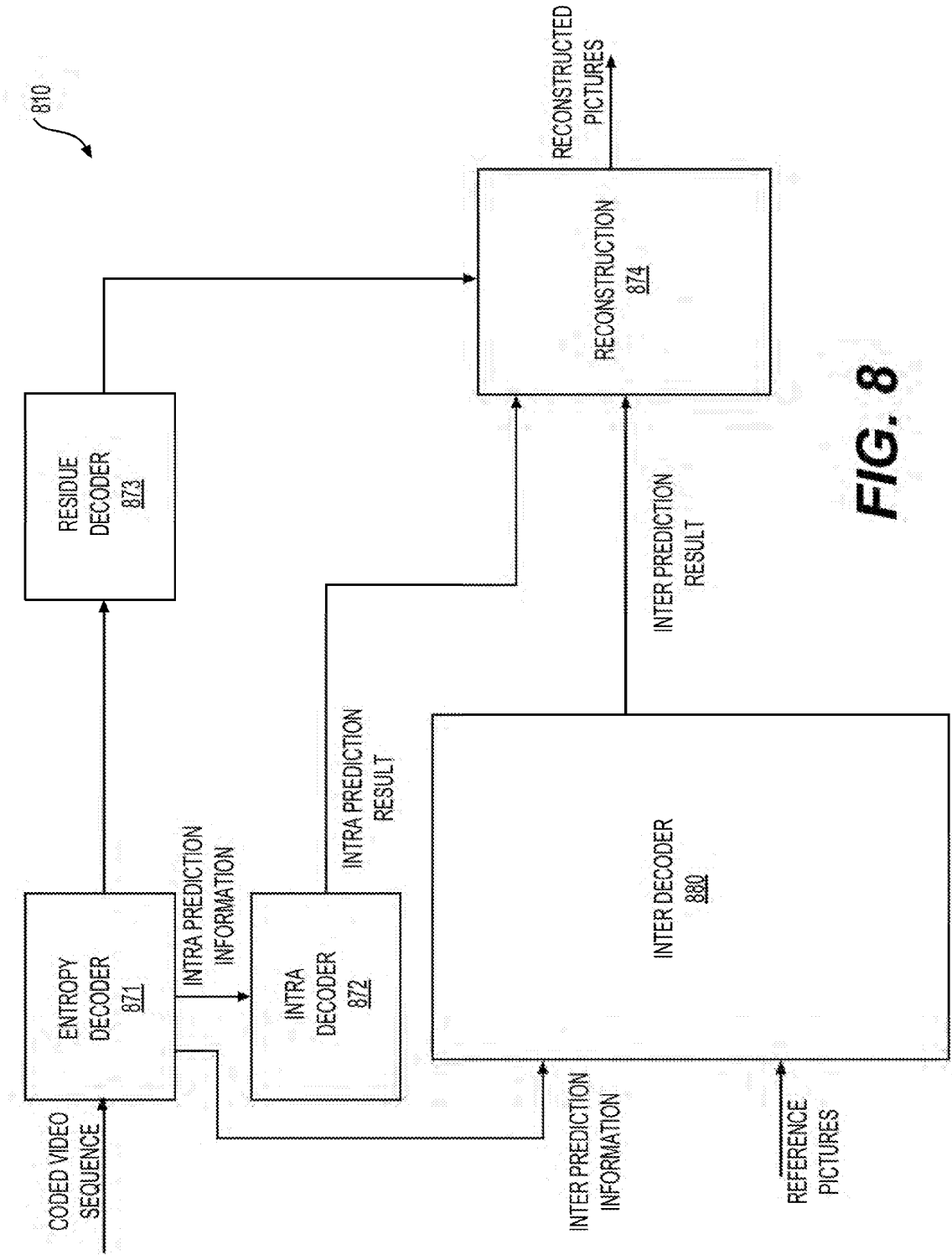


FIG. 8

Descriptor	
	transform_unit(x0, y0, tbWidth, tbHeight, treeType) {
	if(treeType == SINGLE_TREE treeType == DUAL_TREE_LUMA)
ac(v)	tu_cbf_luma[x0 y0]
	if(treeType == SINGLE_TREE treeType == DUAL_TREE_CHROMA) {
ac(v)	tu_cbf_cb[x0 y0]
ac(v)	tu_cbf_cr[x0 y0]
	}
	if(
	((CuPredMode[x0 y0] == MODE_INTRA) && sps_mts_intra_enabled_flag)
	((CuPredMode[x0 y0] == MODE_INTER) && sps_mts_inter_enabled_flag)
	&& tu_cbf_luma[x0 y0] && treeType != DUAL_TREE_CHROMA
	&& (tbWidth <= 32) && (tbHeight <= 32))
ac(v)	cu_mts_flag[x0 y0]
	if(tu_cbf_luma[x0 y0])
	residual_coding(x0, y0, log2(tbWidth), log2(tbHeight), 0)
	if(tu_cbf_cb[x0 y0])
	residual_coding(x0, y0, log2(tbWidth / 2), log2(tbHeight / 2), 1)
	if(tu_cbf_cr[x0 y0])
	residual_coding(x0, y0, log2(tbWidth / 2), log2(tbHeight / 2), 2)
	}

FIG. 9

Descriptor	
	residual_coding(x0, y0, log2TbWidth, log2TbHeight, cldx) {
	if(transform_skip_enabled_flag && (cldx != 0 cu_rmts_flag[x0][y0] == 0) &&
	(log2TbWidth <= 2) && (log2TbHeight <= 2))
	transform_skip_flag[x0][y0][cldx]
ae(v)	last_sig_coeff_x_prefix
ae(v)	last_sig_coeff_y_prefix
ae(v)	if(last_sig_coeff_x_prefix > 3)
	last_sig_coeff_x_suffix
ae(v)	if(last_sig_coeff_y_prefix > 3)
	last_sig_coeff_y_suffix

FIG. 10A

Continue from FIG. 10A

if(dep_quant_enabled_flag) {	
QState = startQStateSb	
for(n = numSbCoeff - 1; n >= 0; n- -) {	
xC = (xS << log2SbSize) +	
DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) +	
DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(sig_coeff_flag[xC][yC])	
TransCoeffLevel[x0][y0][cIdx][xC][yC] =	
(2 * AbsLevel[xC][yC] - (QState > 1 ? 1 : 0)) *	
(1 - 2 * coeff_sign_flag[n])	
QState = QStateTransTable[QState][par_level_flag[n]]	
} else {	
sumAbsLevel = 0	
for(n = numSbCoeff - 1; n >= 0; n- -) {	
xC = (xS << log2SbSize) +	
DiagScanOrder[log2SbSize][log2SbSize][n][0]	
yC = (yS << log2SbSize) +	
DiagScanOrder[log2SbSize][log2SbSize][n][1]	
if(sig_coeff_flag[xC][yC]) {	

FIG. 10B

Continue from FIG. 10B

TransCoeffLevel[x0][y0][cldx][xC][yC] =	
AbsLevel[xC][yC] * (1 - 2 * coeff_sign_flag[n])	
if(signHidden) {	
sumAbsLevel += AbsLevel[xC][yC]	
if(n == firstSigScanPosSb) && (sumAbsLevel % 2) == 1)	
TransCoeffLevel[x0][y0][cldx][xC][yC] =	
- TransCoeffLevel[x0][y0][cldx][xC][yC]	
}	
}	
}	
}	
}	
if(cu_mis_flag[x0][y0] && (cldx == 0) &&	
transform_skip_flag[x0][y0][cldx] &&	
((CuPredMode[x0][y0] == MODE_INTRA && numSigCoeff > 2)	
(CuPredMode[x0][y0] == MODE_INTER)) {	
mis_idx[x0][y0]	act()
}	

FIG. 10C

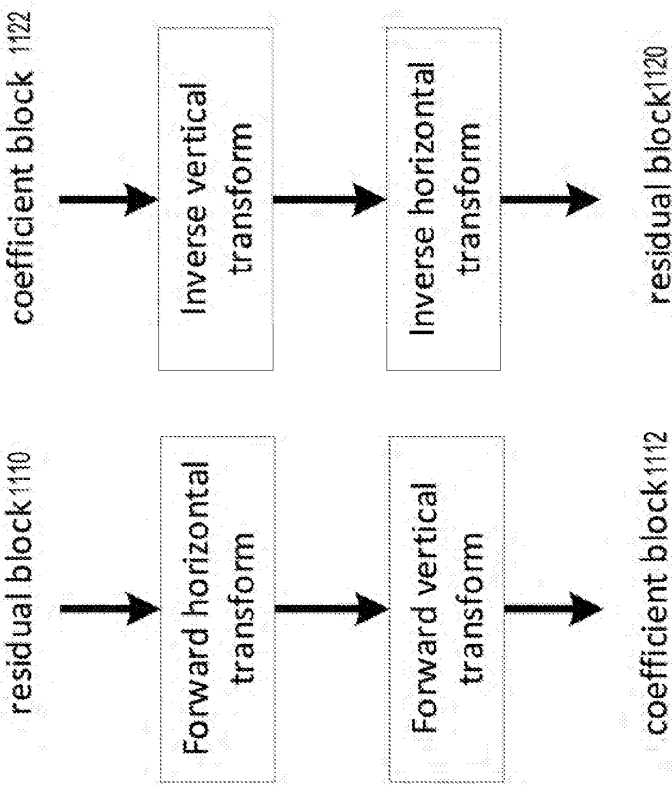


FIG. 11A

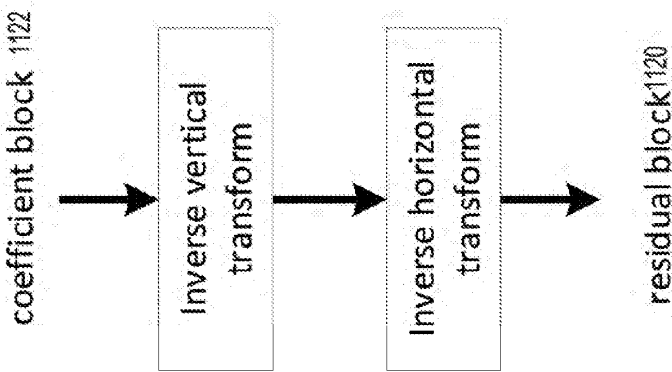


FIG. 11B

Inputs to this process are:

- a luma location ($xTbY$, $yTbY$) specifying the top-left sample of the current luma transform block relative to the top-left luma sample of the current picture,
- a variable $nTbW$ specifying the width of the current transform block,
- a variable $nTbH$ specifying the height of the current transform block,
- a variable $cldx$ specifying the colour component of the current block,
- an $(nTbW) \times (nTbH)$ array $d[x][y]$ of scaled transform coefficients with $x = 0..nTbW - 1$, $y = 0..nTbH - 1$.

Output of this process is the $(nTbW) \times (nTbH)$ array $r[x][y]$ of residual samples with $x = 0..nTbW - 1$, $y = 0..nTbH - 1$.

FIG. 12A

Continue from FIG. 12A

The variable `implicitMtsEnabled` is derived as follows:

- If `sps_mts_enabled_flag` is equal to 1 and one of the following conditions is true, `implicitMtsEnabled` is set equal to 1:
 - `IntraSubPartitionsSplitType` is not equal to `ISP_NO_SPLIT`
 - `cu_sbt_flag` is equal to 1 and `Max( nTbW, nTbH )` is less than or equal to 32
 - `sps_explicit_mts_intra_enabled_flag` and `sps_explicit_mts_inter_enabled_flag` are both equal to 0 and `CuPredMode[ xTbY ][ yTbY ]` is equal to `MODE_INTRA`

Otherwise, `implicitMtsEnabled` is set equal to 0.

FIG. 12B

Continue from FIG. 12B

The variable `trTypeHor` specifying the horizontal transform kernel and the variable `trTypeVer` specifying the vertical transform kernel are derived as follows:

- If `cldx` is greater than 0, `trTypeHor` and `trTypeVer` are set equal to 0.
- Otherwise, if `implicitMtsEnabled` is equal to 1, the following applies:
 - If `IntraSubPartitionsSplitType` is not equal to `ISP_NO_SPLIT`, `trTypeHor` and `trTypeVer` are specified depending on `intraPredMode`.
 - Otherwise, if `cu_sbt_flag` is equal to 1, `trTypeHor` and `trTypeVer` are specified depending on `cu_sbt_horizontal_flag` and `cu_sbt_pos_flag`.
 - Otherwise (`sps_explicit_mts_intra_enabled_flag` and `sps_explicit_mts_inter_enabled_flag` are equal to 0), `trTypeHor` and `trTypeVer` are derived as follows:

$$\text{trTypeHor} = (\text{nTbW} \geq 4 \ \&\& \ \text{nTbW} \leq 16 \ \&\& \ \text{nTbW} \leq \text{nTbH}) ? 1 : 0$$

$$\text{trTypeVer} = (\text{nTbH} \geq 4 \ \&\& \ \text{nTbH} \leq 16 \ \&\& \ \text{nTbH} \leq \text{nTbW}) ? 1 : 0$$
FIG. 12C

Continue from FIG. 12C

... Otherwise, trTypeHor and trTypeVer are specified depending on tu_mts_idx[xTbY][yTbY].

The variables nonZeroW and nonZeroH are derived as follows:

$$\text{nonZeroW} = \text{Min}(\text{nTbW}, (\text{trTypeHor} > 0) ? 16 : 32)$$

$$\text{nonZeroH} = \text{Min}(\text{nTbH}, (\text{trTypeVer} > 0) ? 16 : 32)$$

FIG. 12D

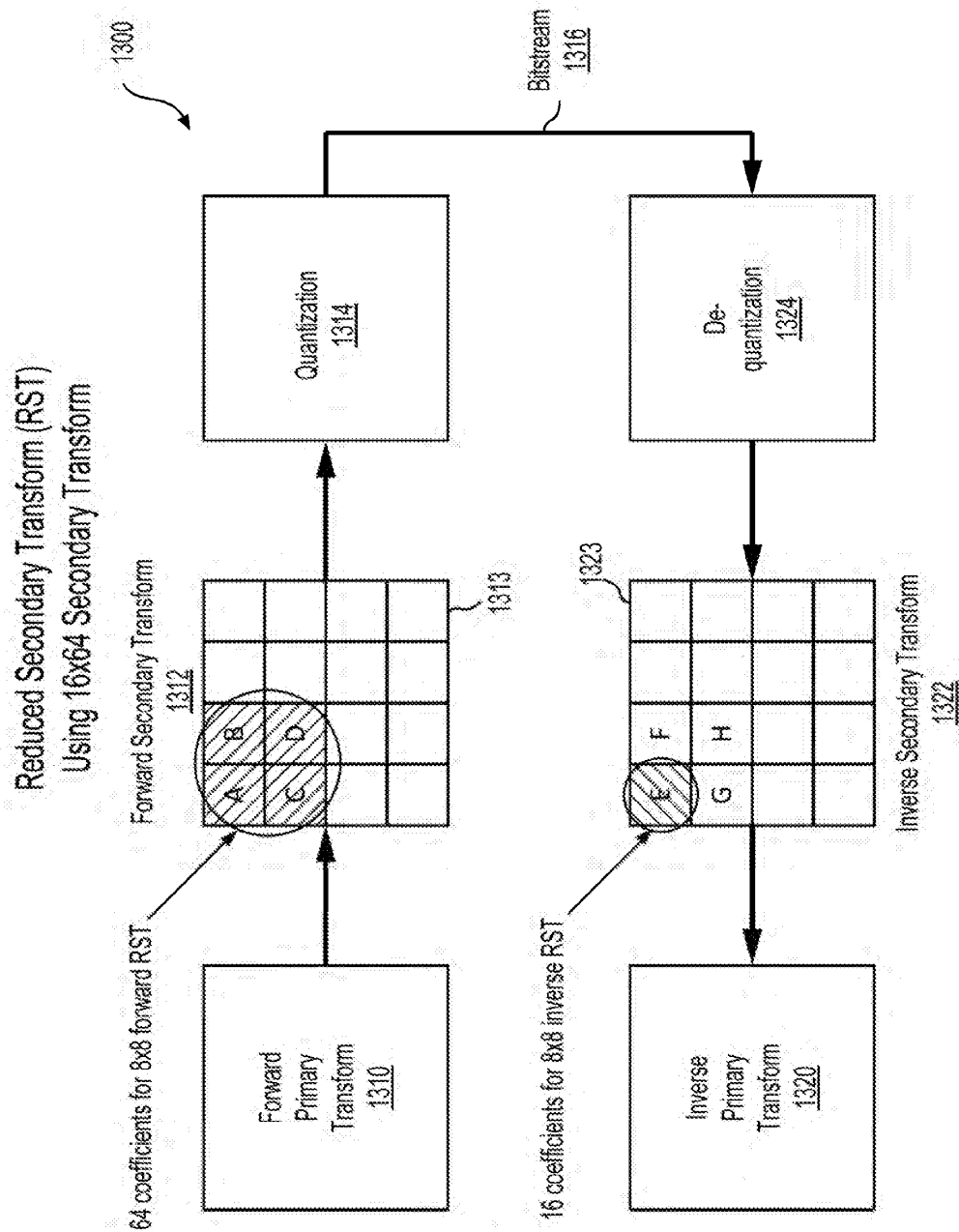
Continue from FIG. 12D

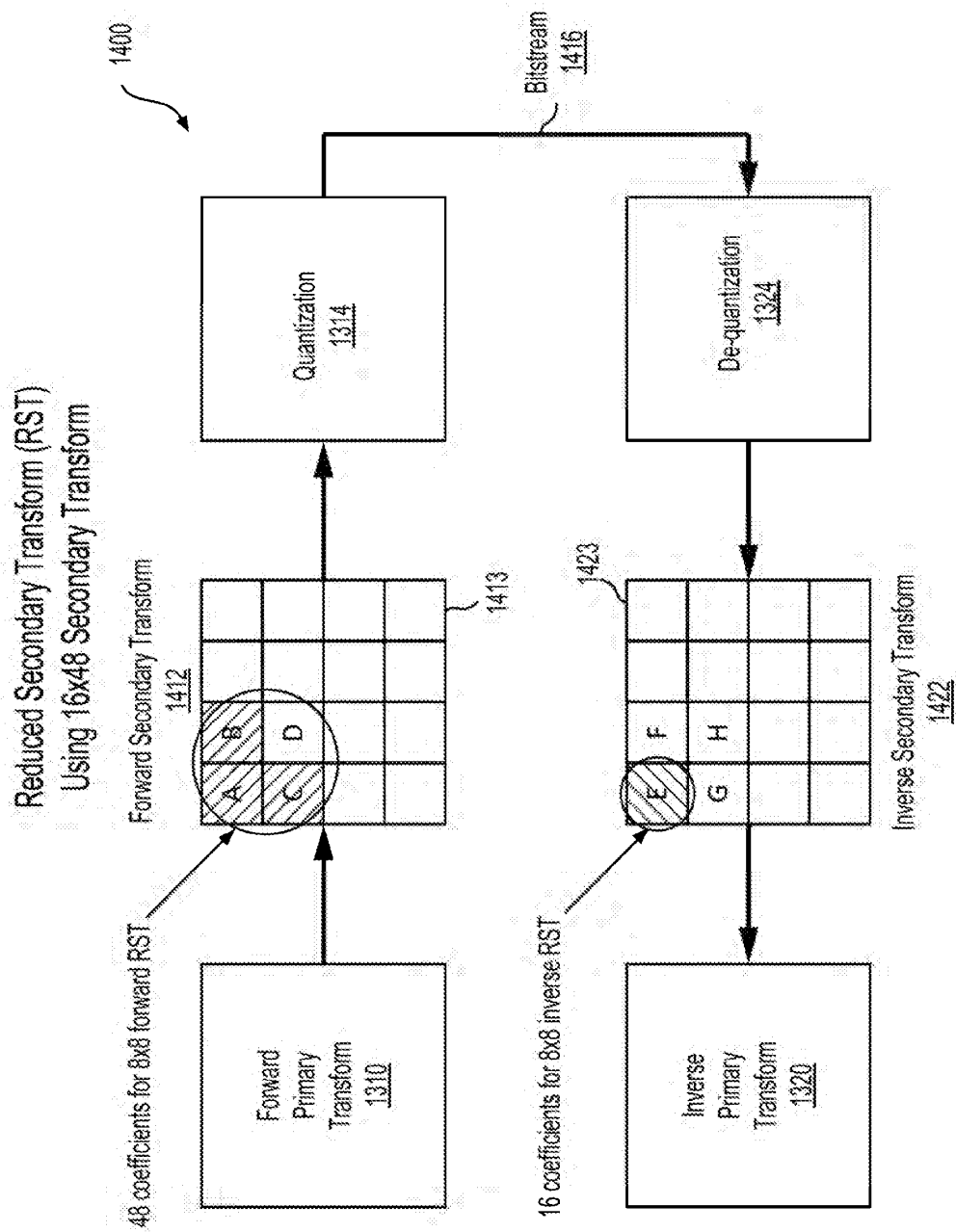
The $(nTbW) \times (nTbH)$ array r of residual samples is derived as follows:

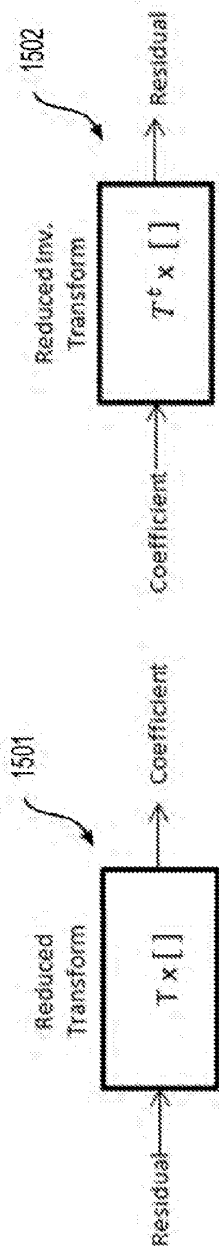
1. When $nTbH$ is greater than 1, each (vertical) column of scaled transform coefficients $d[x][y]$ with $x = 0..nonZeroW - 1$, $y = 0..nonZeroH - 1$ is transformed to $e[x][y]$ with $x = 0..nonZeroW - 1$, $y = 0..nonZeroH - 1$ by invoking the one-dimensional transformation process for each column $x = 0..nonZeroW - 1$ with the height of the transform block $nTbH$, the non-zero height of the scaled transform coefficients $nonZeroH$, the list $d[x][y]$ with $y = 0..nonZeroH - 1$ and the transform type variable $trType$ set equal to $trTypeVer$ as inputs, and the output is the list $e[x][y]$ with $y = 0..nTbH - 1$.
2. When $nTbH$ and $nTbW$ are both greater than 1, the intermediate sample values $g[x][y]$ with $x = 0..nonZeroW - 1$, $y = 0..nTbH - 1$ are derived as follows:

$$g[x][y] = \text{Clip3}(\text{CoeffMin}, \text{CoeffMax}, (e[x][y] + 64) \gg 7)$$
3. When $nTbW$ is greater than 1, each (horizontal) row of the resulting array $g[x][y]$ with $x = 0..nonZeroW - 1$, $y = 0..nTbH - 1$ is transformed to $r[x][y]$ with $x = 0..nTbW - 1$, $y = 0..nTbH - 1$ by invoking the one-dimensional transformation process for each row $y = 0..nTbH - 1$ with the width of the transform block $nTbW$, the non-zero width of the resulting array $g[x][y]$ $nonZeroW$, the list $g[x][y]$ with $x = 0..nonZeroW - 1$ and the transform type variable $trType$ set equal to $trTypeHor$ as inputs, and the output is the list $r[x][y]$ with $x = 0..nTbW - 1$.

FIG. 12E







Forward and Inverse Reduced Transform

FIG. 15A

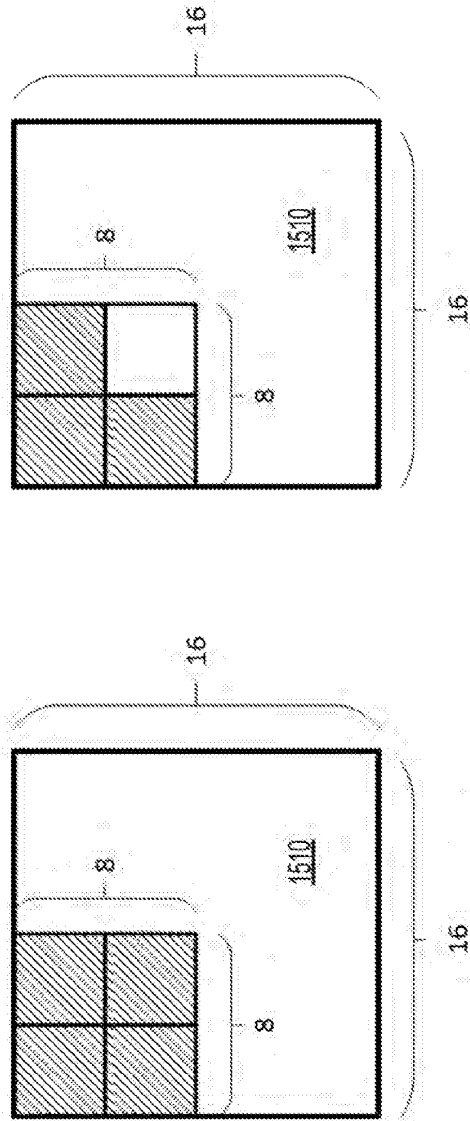


FIG. 15B

FIG. 15C

1550



Transform Set Selection Table

IntraPredMode	Tr. set index
IntraPredMode < 0	1
0 <= IntraPredMode <= 1	0
2 <= IntraPredMode <= 12	1
13 <= IntraPredMode <= 23	2
24 <= IntraPredMode <= 44	3
45 <= IntraPredMode <= 55	2
56 <= IntraPredMode	1

FIG. 15D

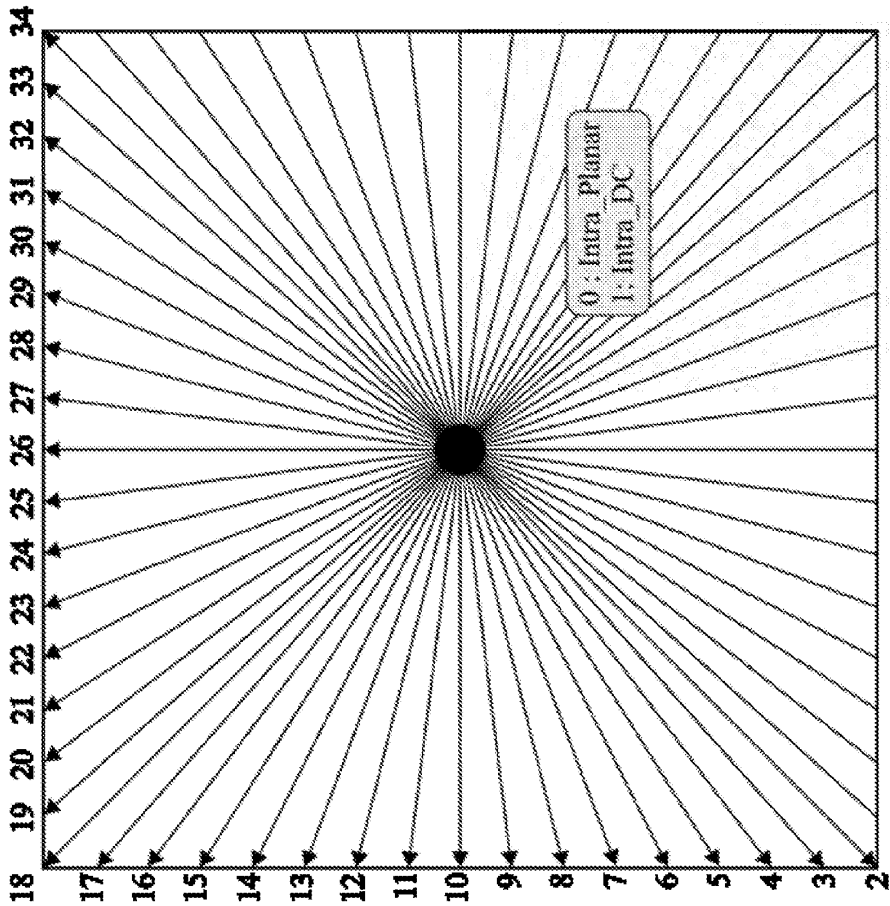


FIG. 16A

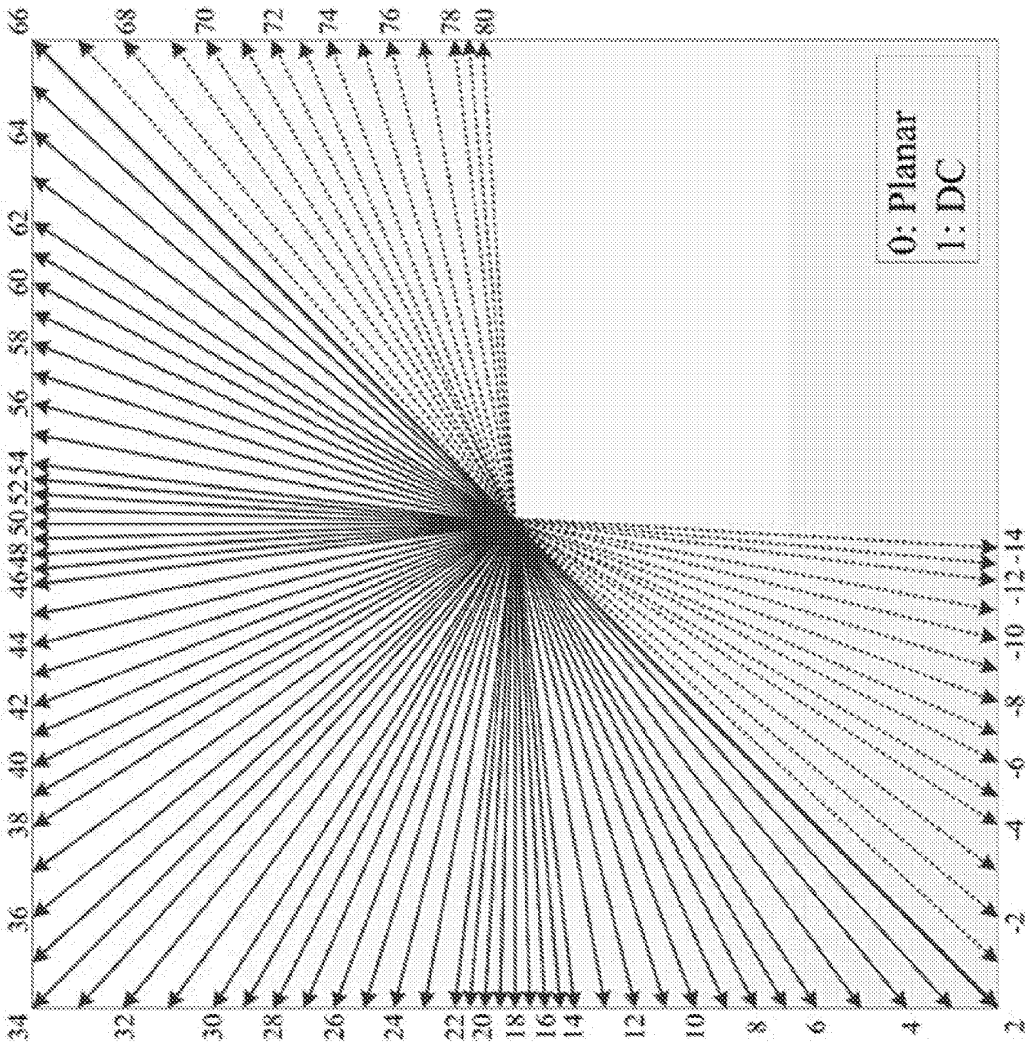


FIG. 16B

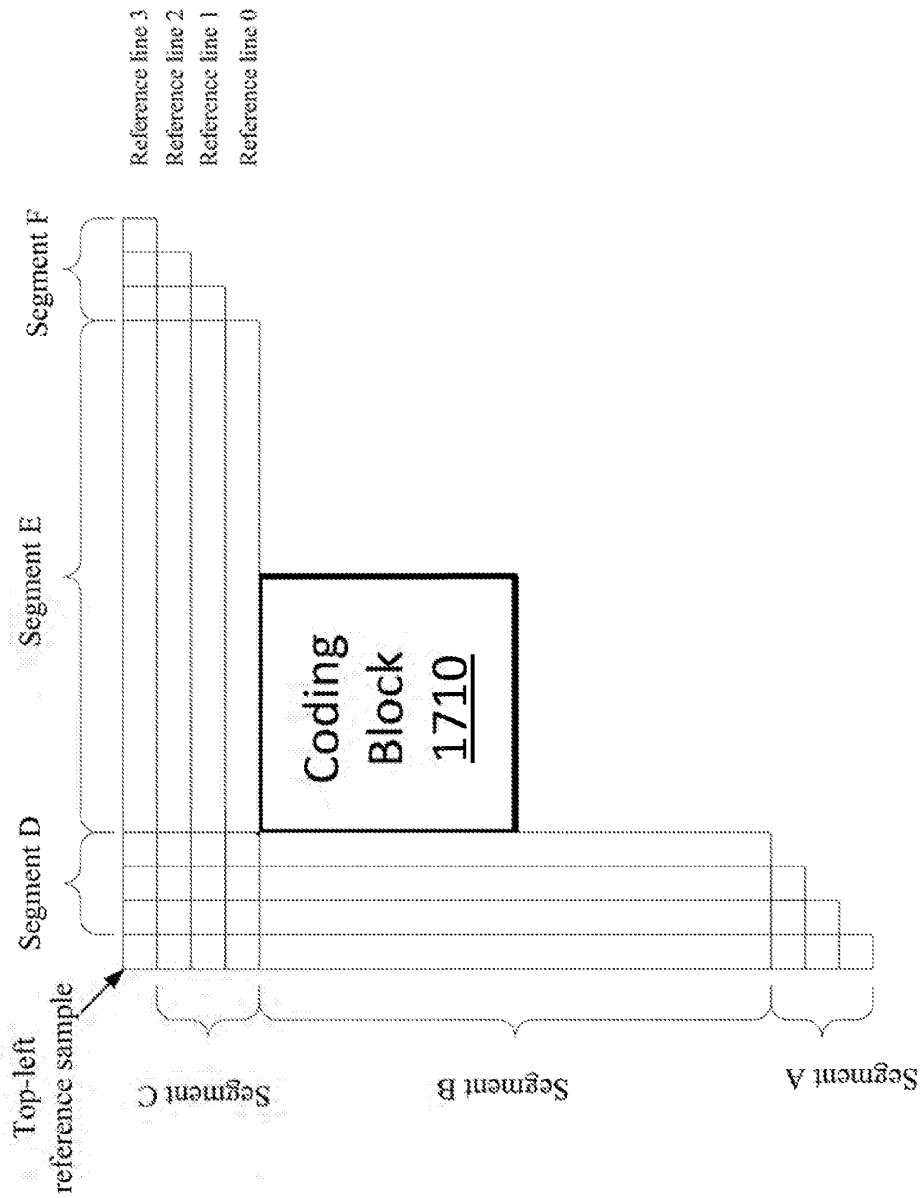


FIG. 17

TABLE 4: NUMBER OF SUB-PARTITIONS THAT DEPENDS ON BLOCK SIZE

BLOCK SIZE	NUMBER OF SUB-PARTITIONS
4X4	NOT DIVIDED
4X8 AND 8X4	2
ALL OTHER CASES	4

FIG. 18

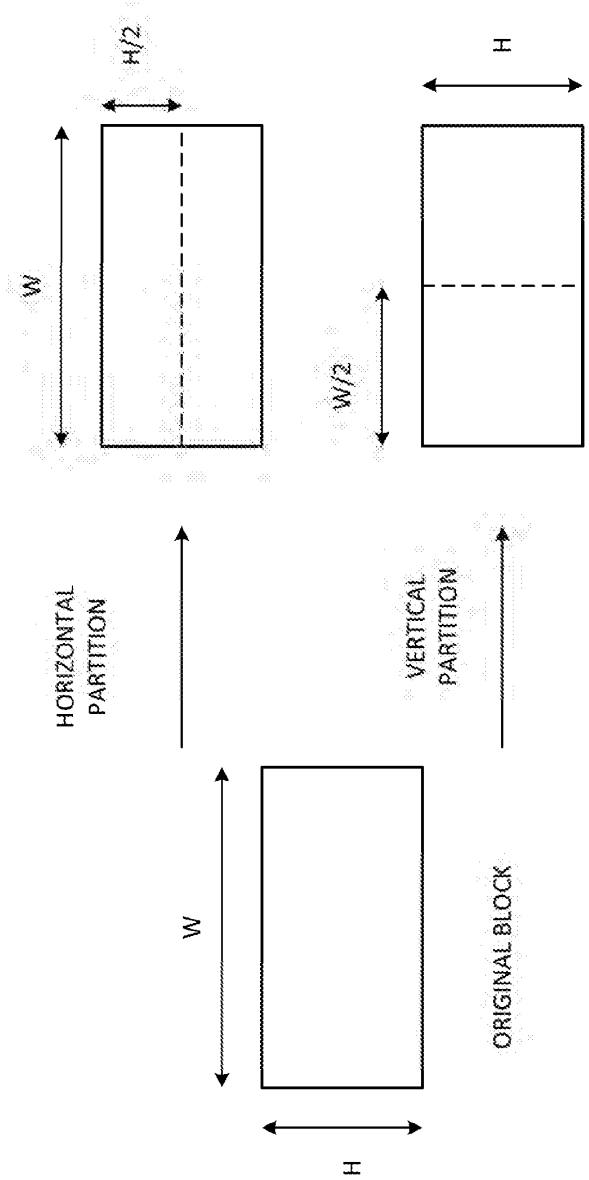


FIG. 19

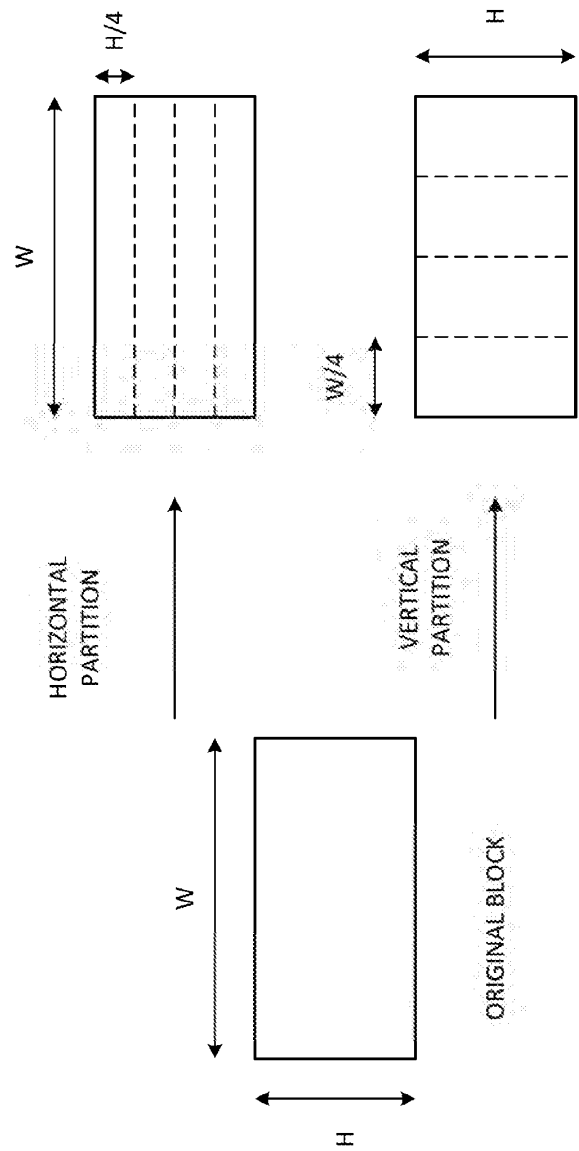


FIG. 20

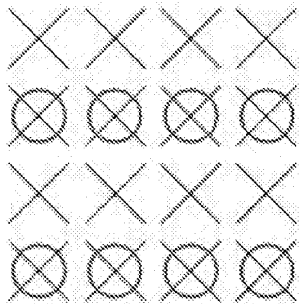
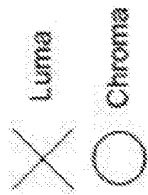


FIG. 21A

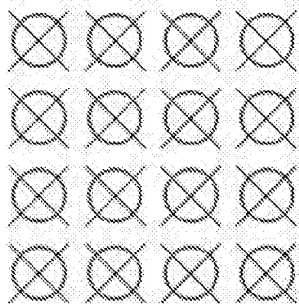


FIG. 21B

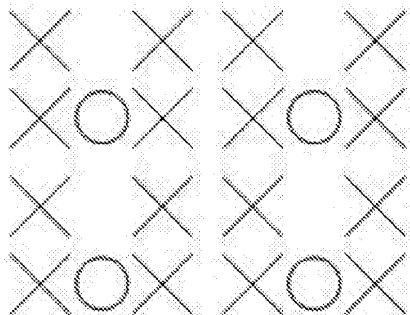


FIG. 21C

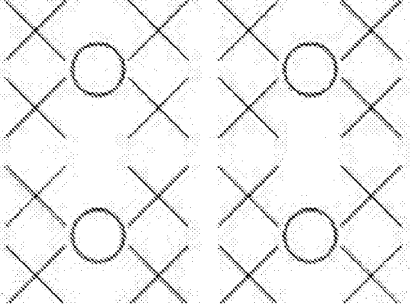


FIG. 21D

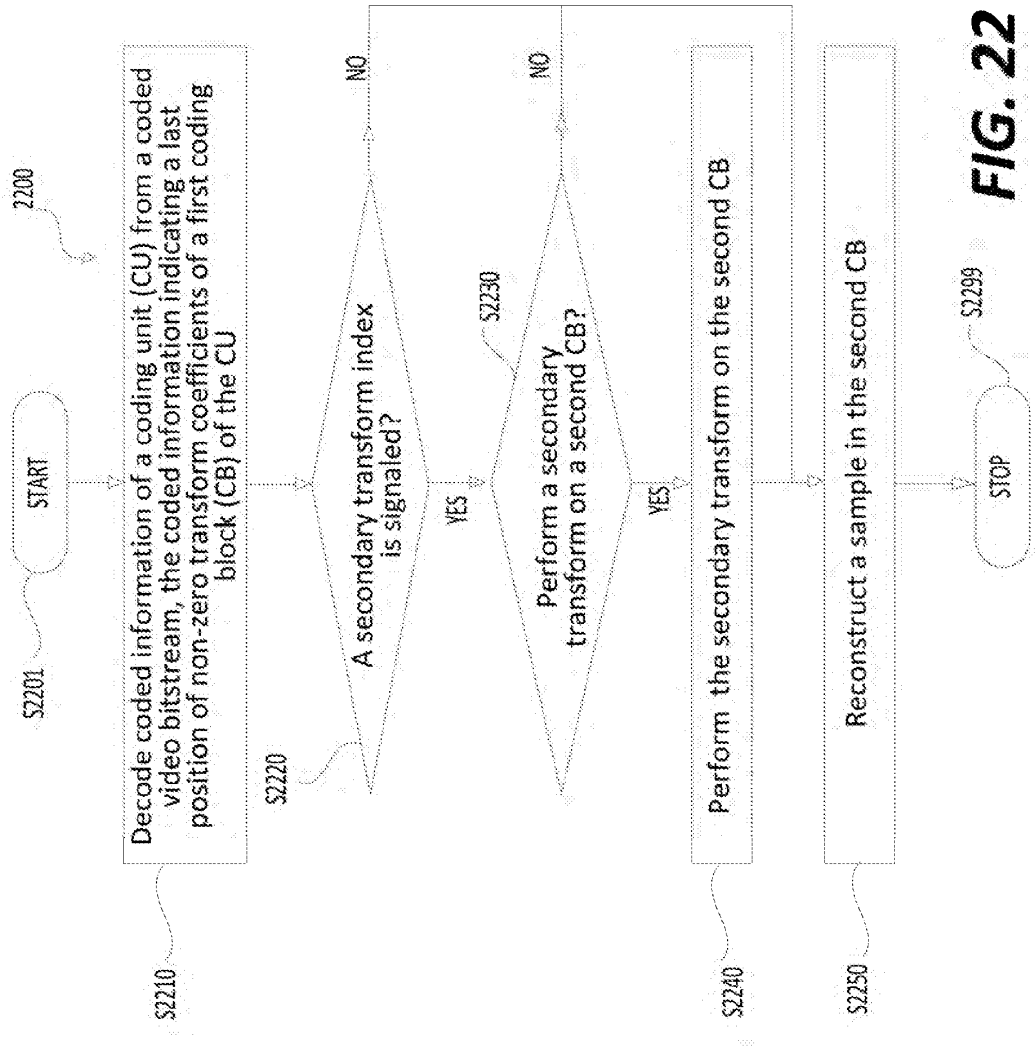


FIG. 22

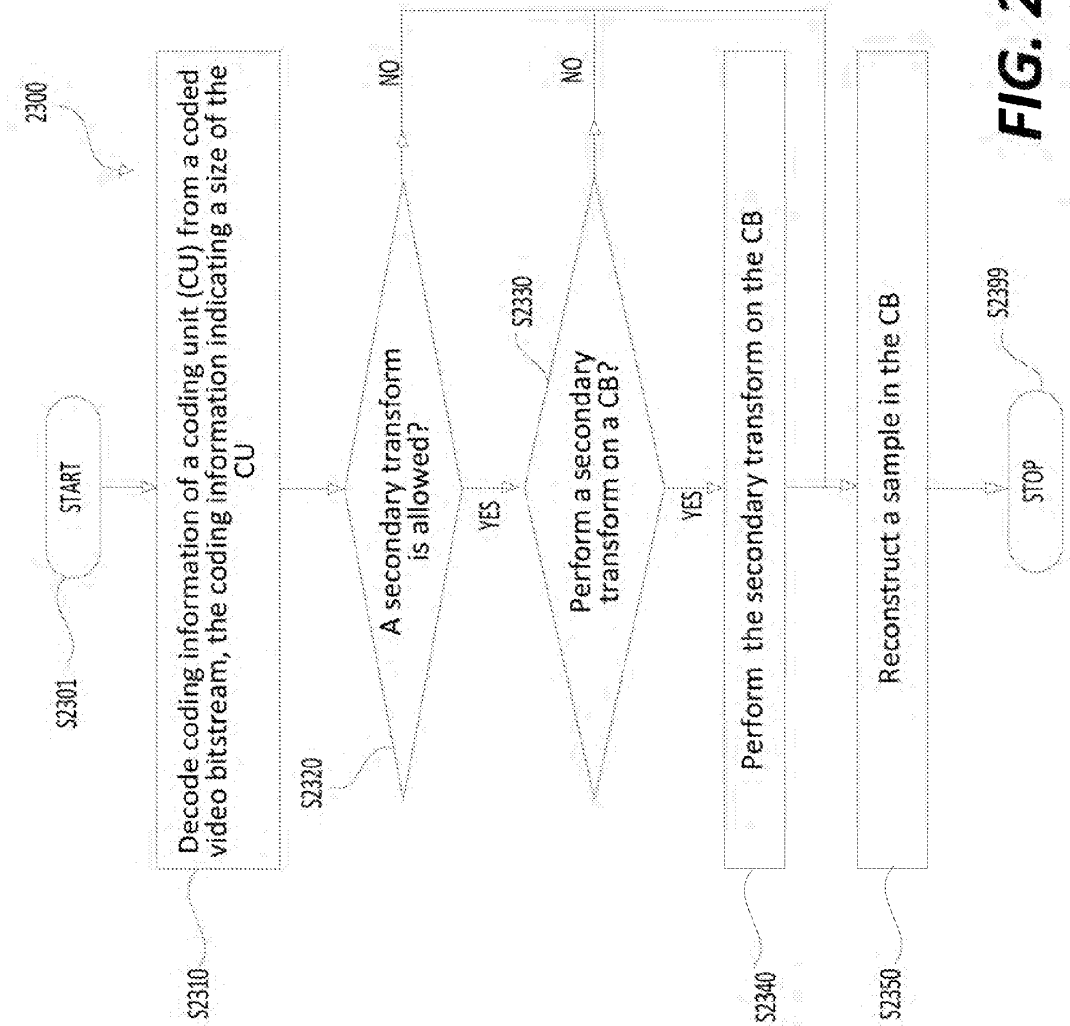


FIG. 23

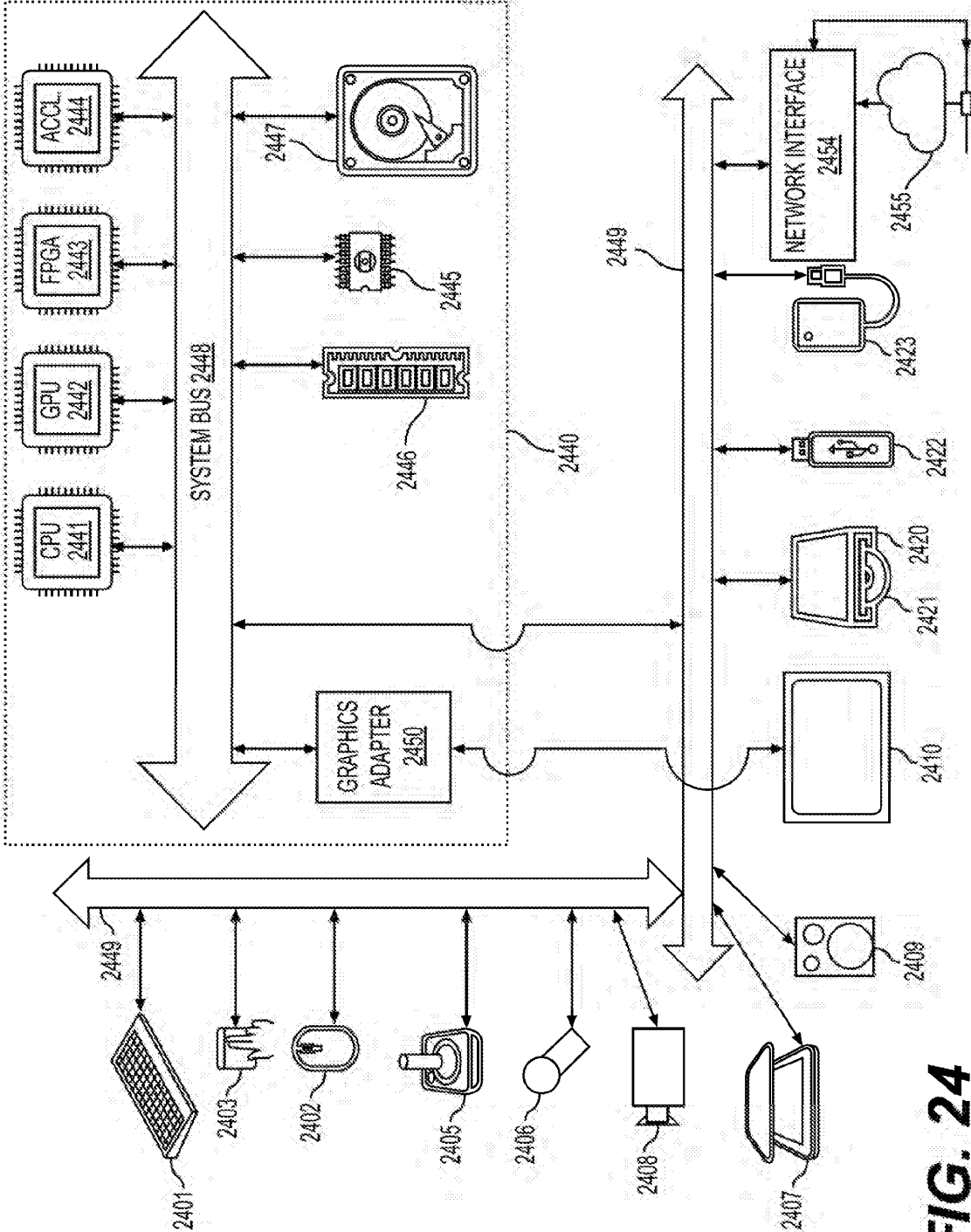


FIG. 24