

FIG. 1

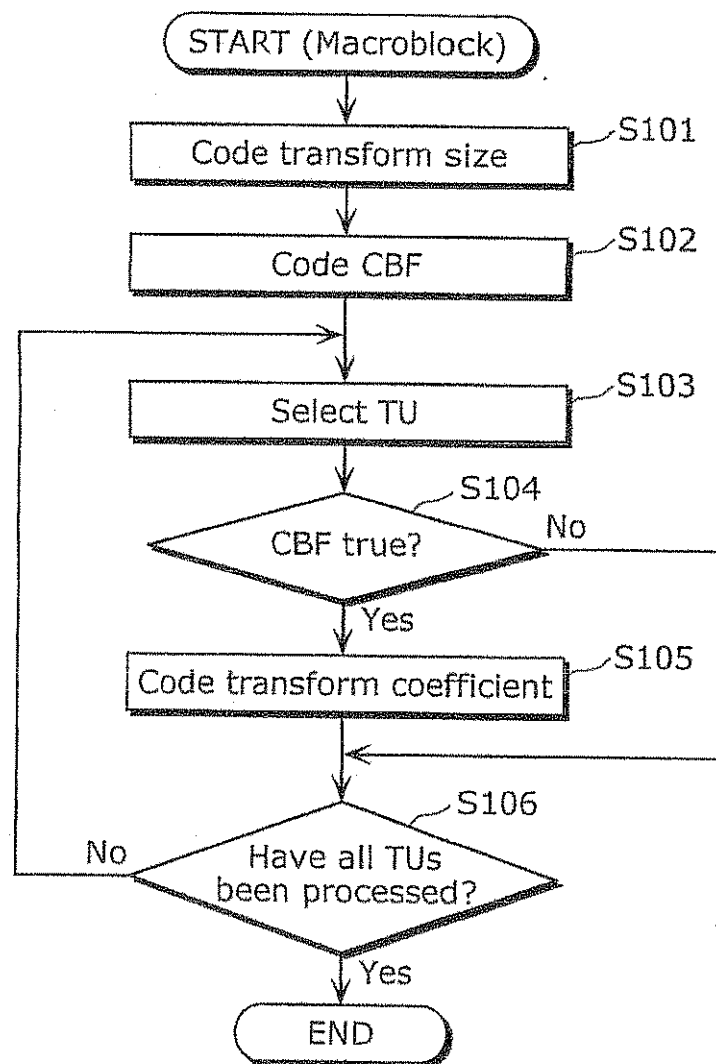


FIG. 2

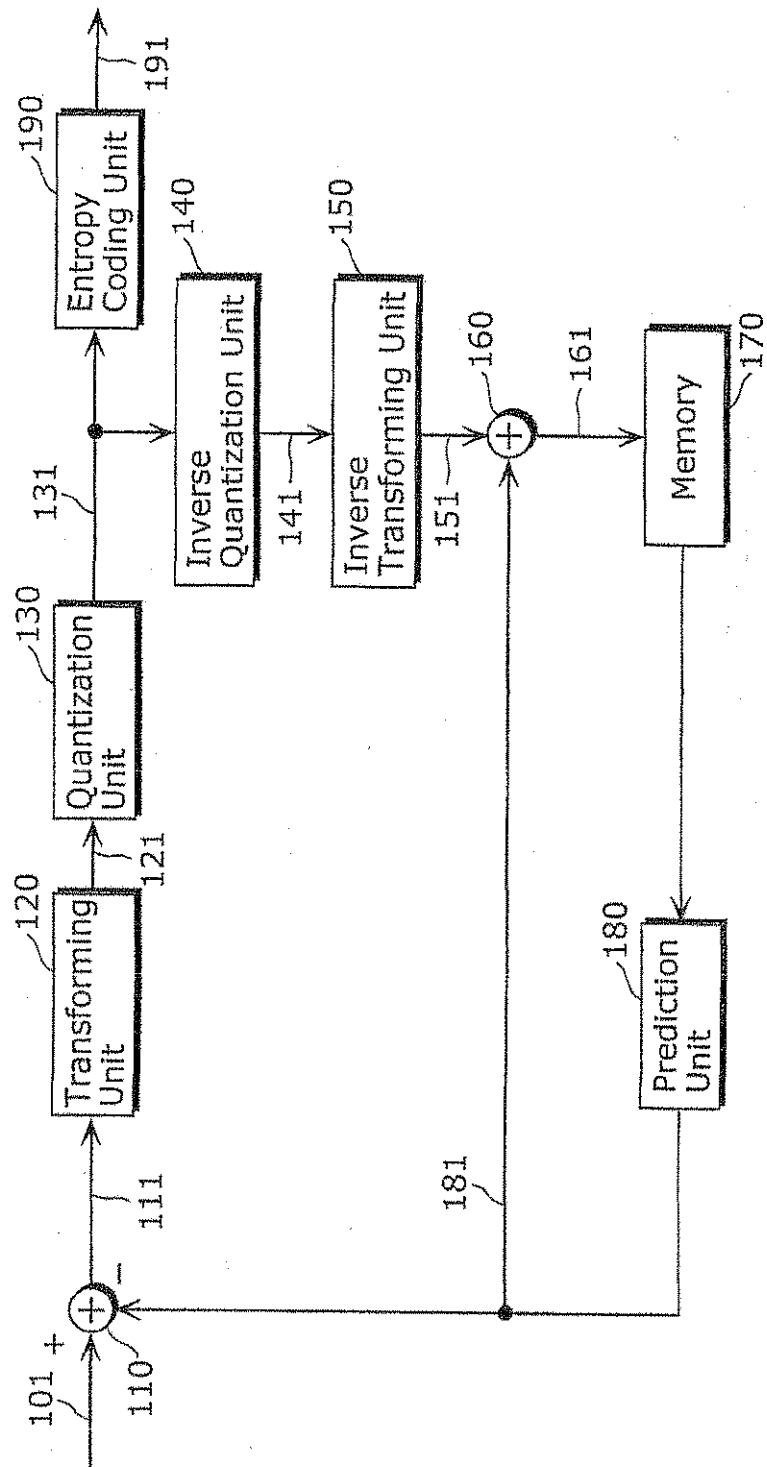


FIG. 3

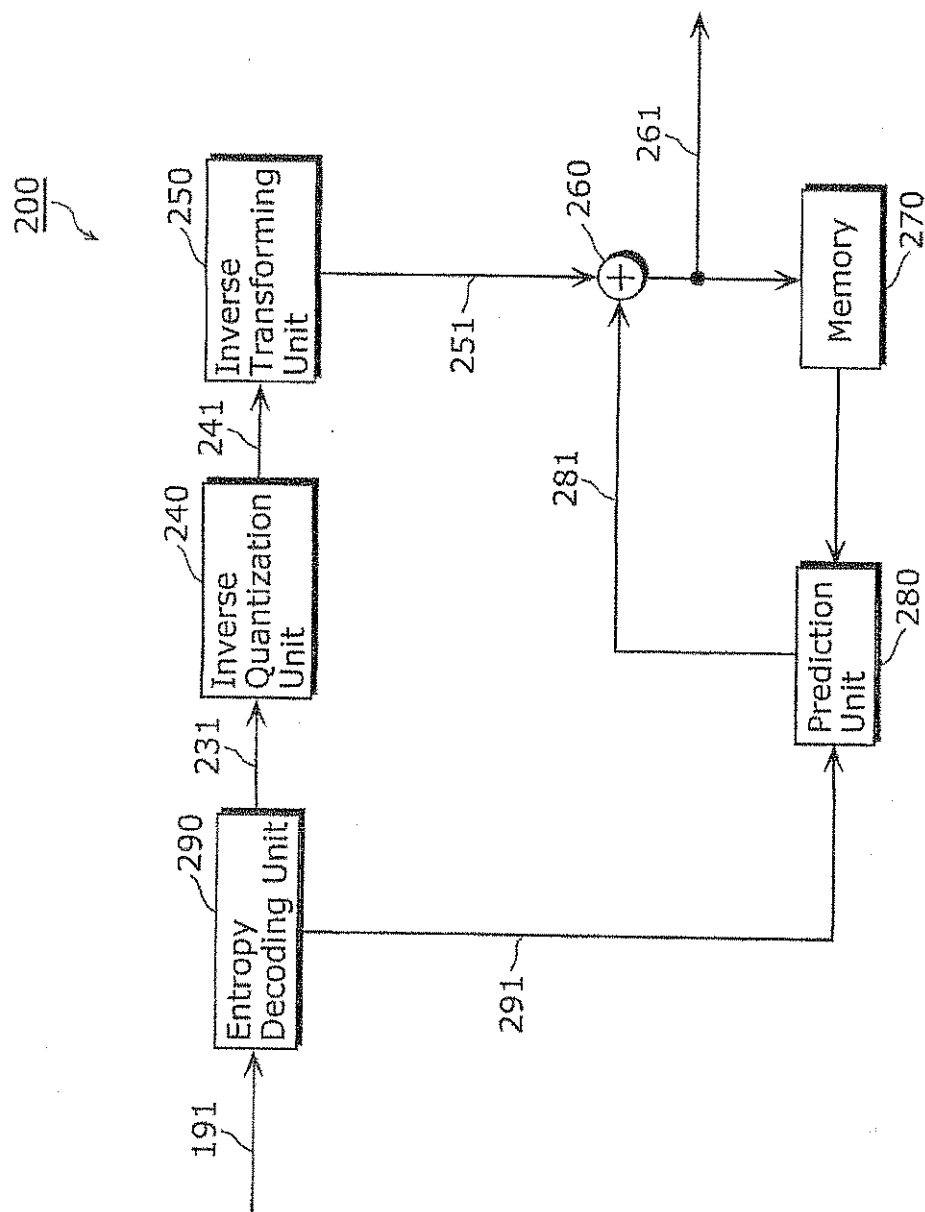


FIG. 4A

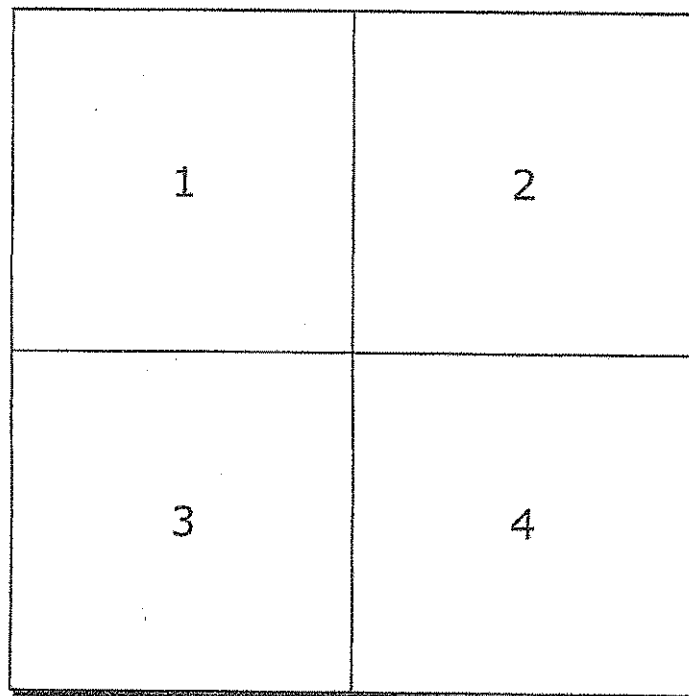


FIG. 4B

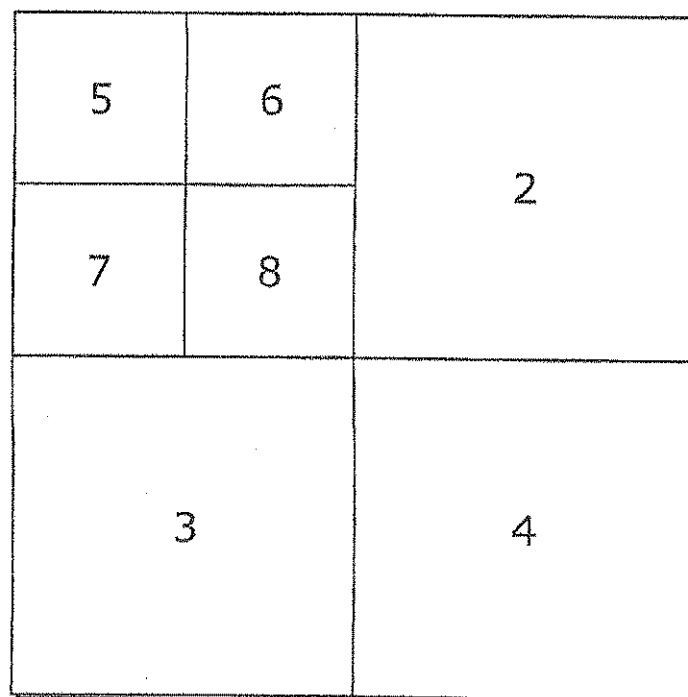


FIG. 5

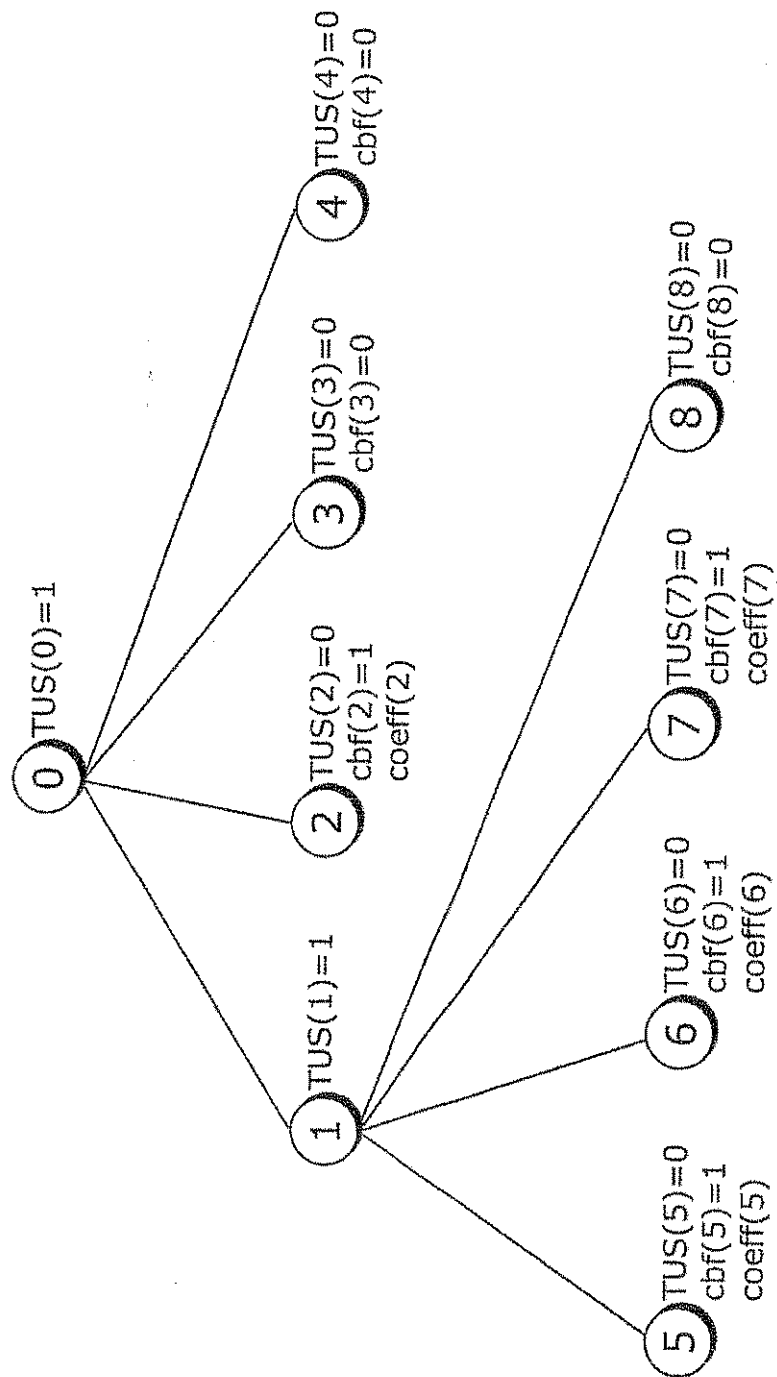


FIG. 6

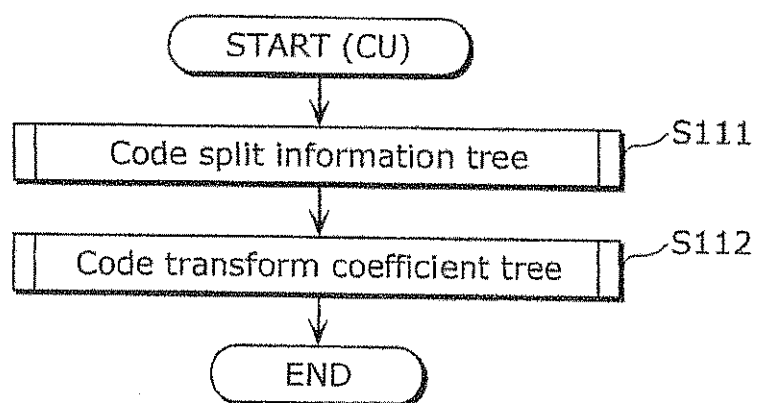


FIG. 7

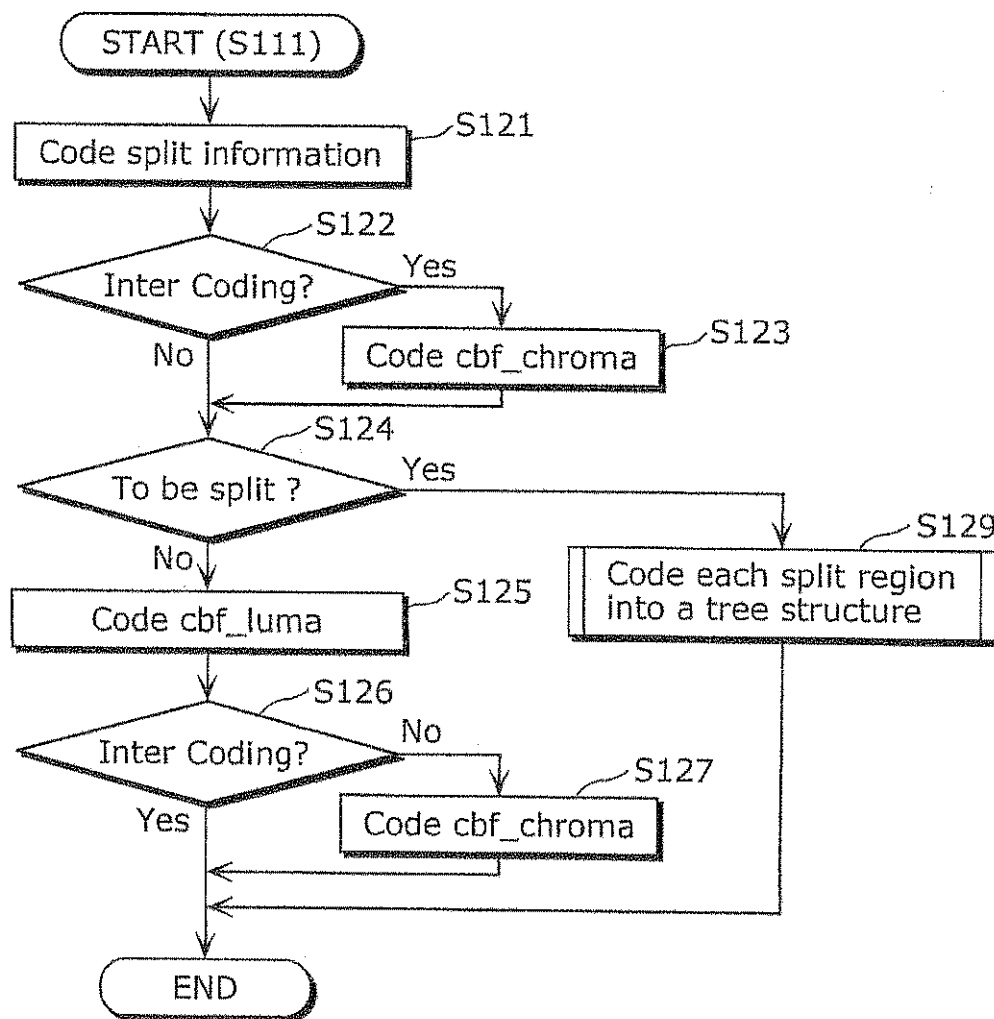


FIG. 8

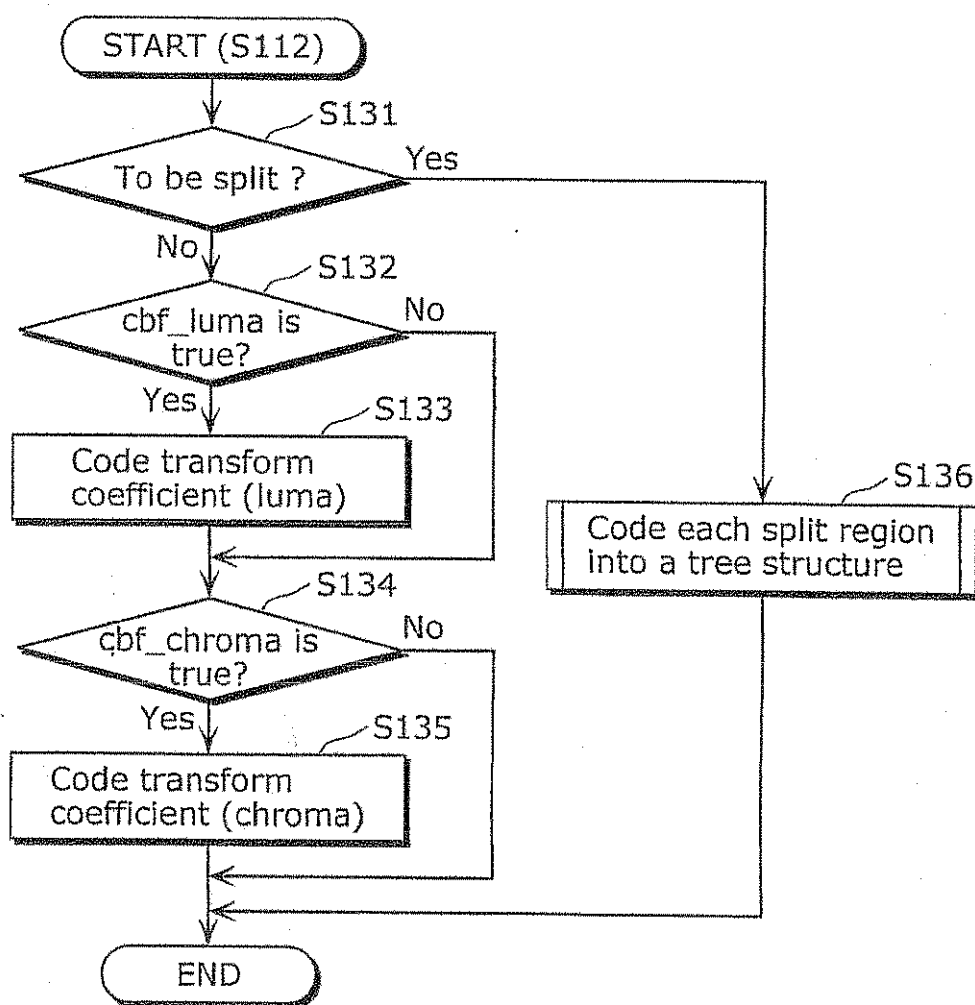


FIG. 9

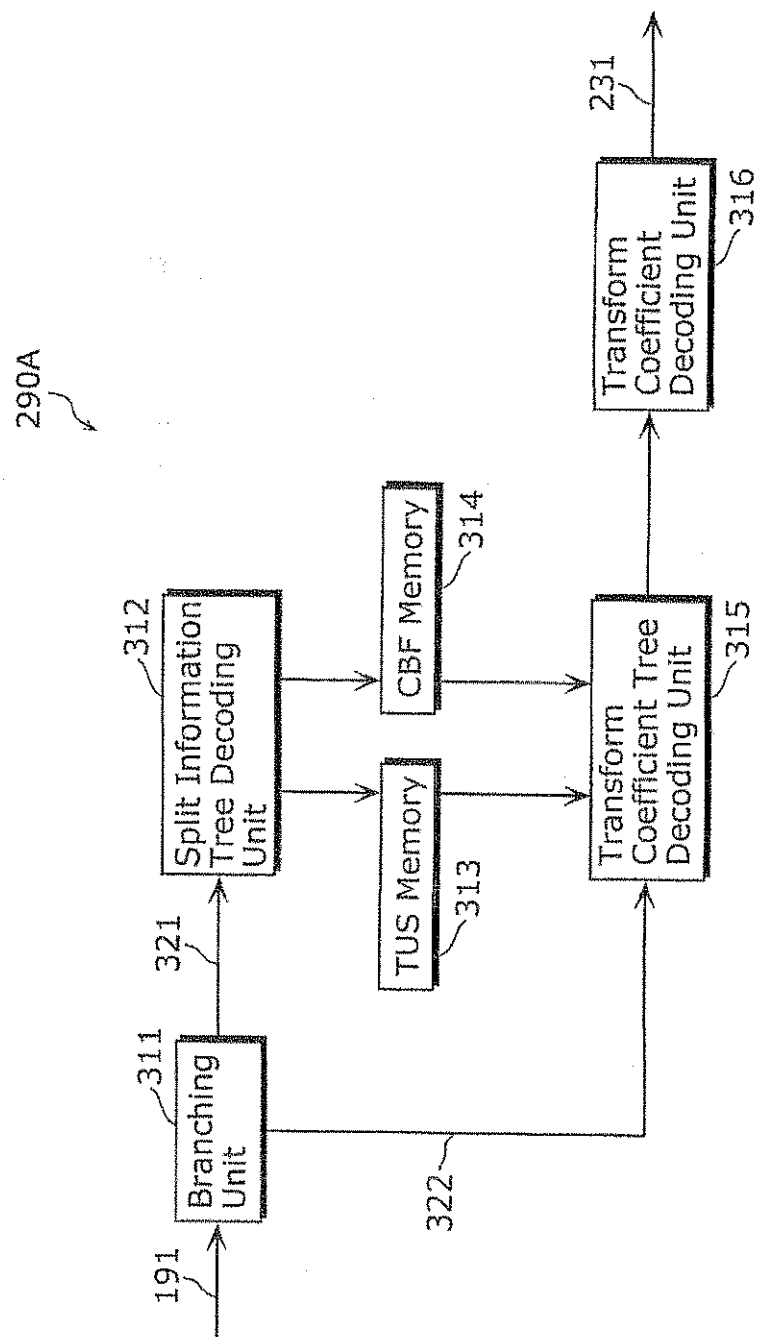


FIG. 10A

Split Information Tree	Transform Coefficient Tree
$TUS(0)=1, TUS(1)=1, TUS(5)=0, cbf(5)=1, TUS(6)=0, cbf(6)=1, \dots$	$coeff(5), coeff(6), \dots$

FIG. 10B

Transform Unified Tree
$TUS(0)=1, TUS(1)=1, TUS(5)=0, cbf(5)=1, coeff(5), TUS(6)=0, cbf(6)=1, coeff(6), \dots$

FIG. 11

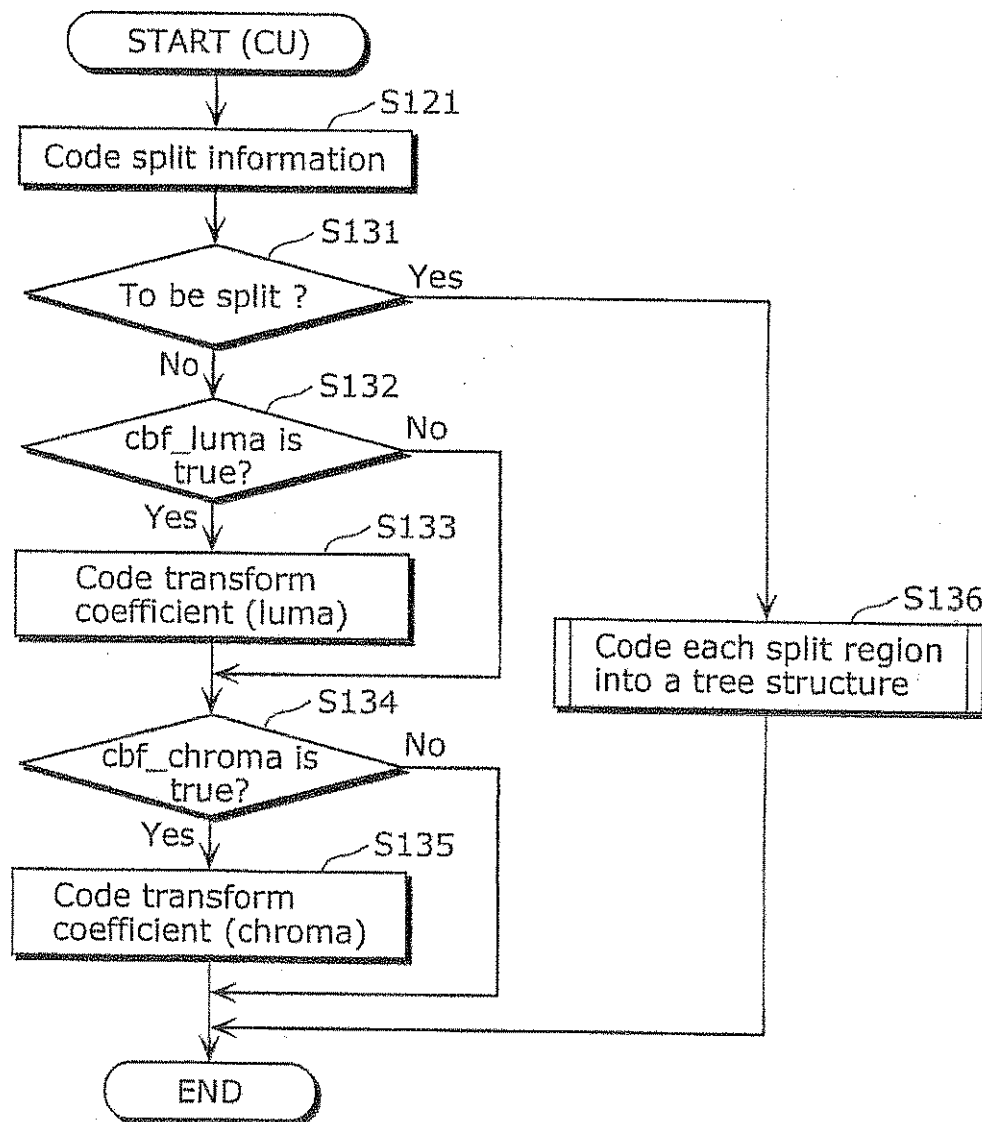


FIG. 12A

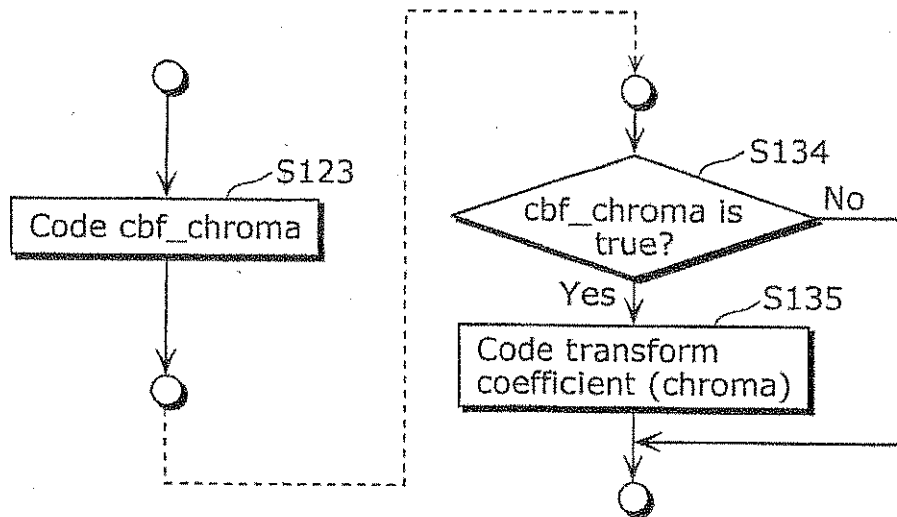


FIG. 12B

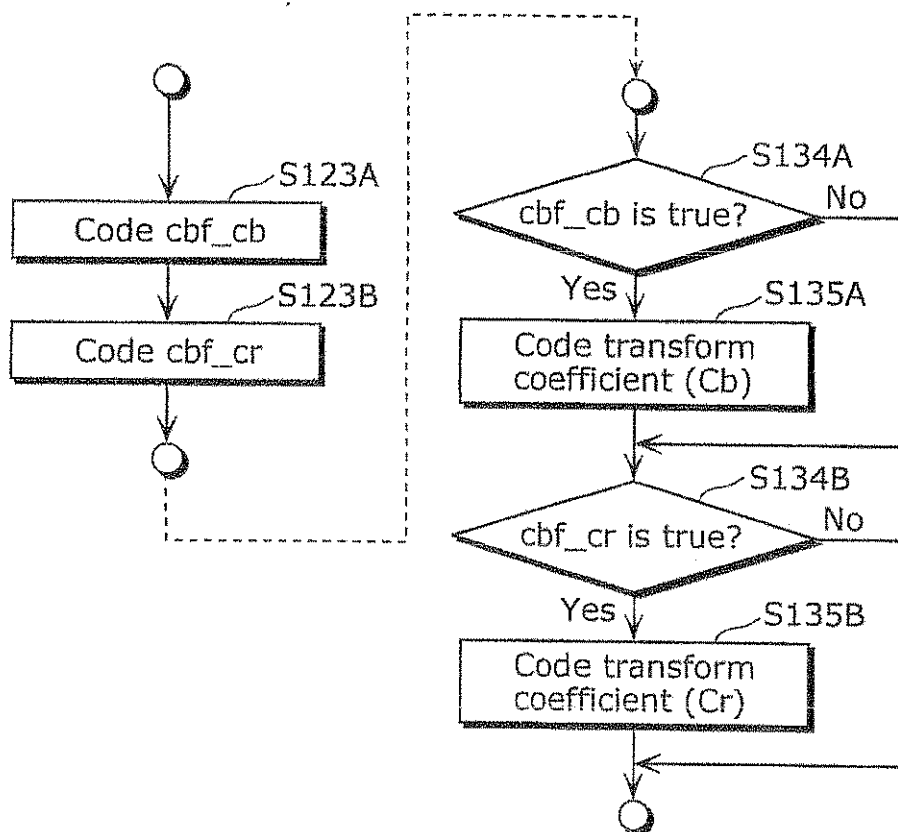


FIG. 13

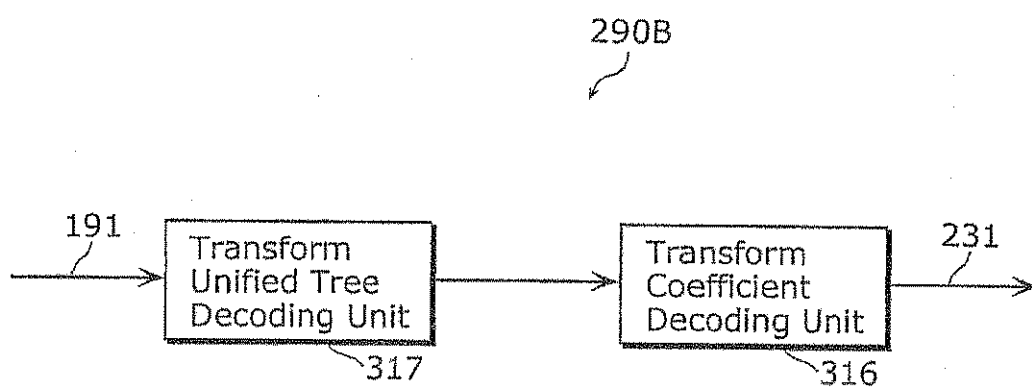


FIG. 14A

cbf(0)	cbf(1)
cbf(2)	cbf(3)

FIG. 14B

cbf(0) =0	cbf(1) =0
cbf(2) =0	cbf(3) →1

FIG. 14C

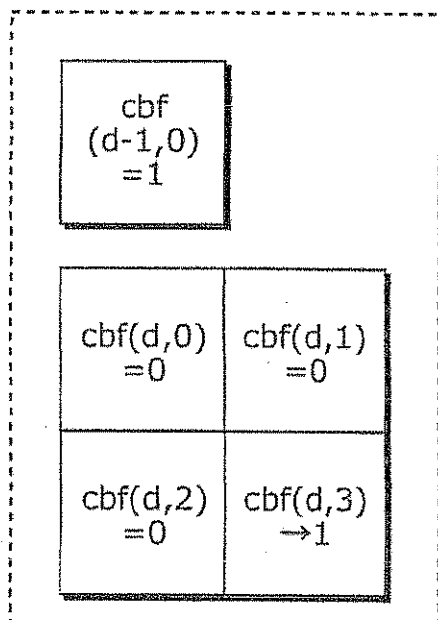


FIG. 14D

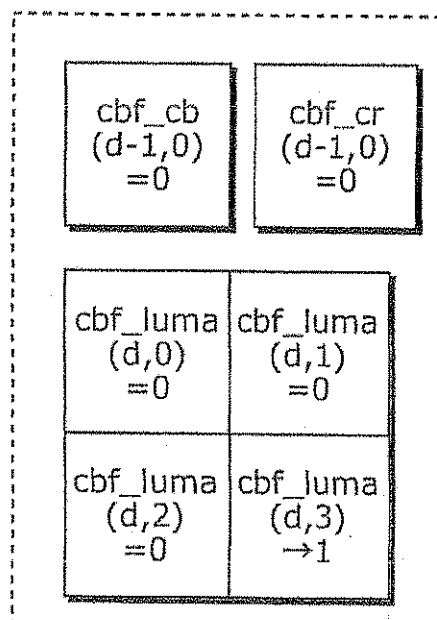


FIG. 15

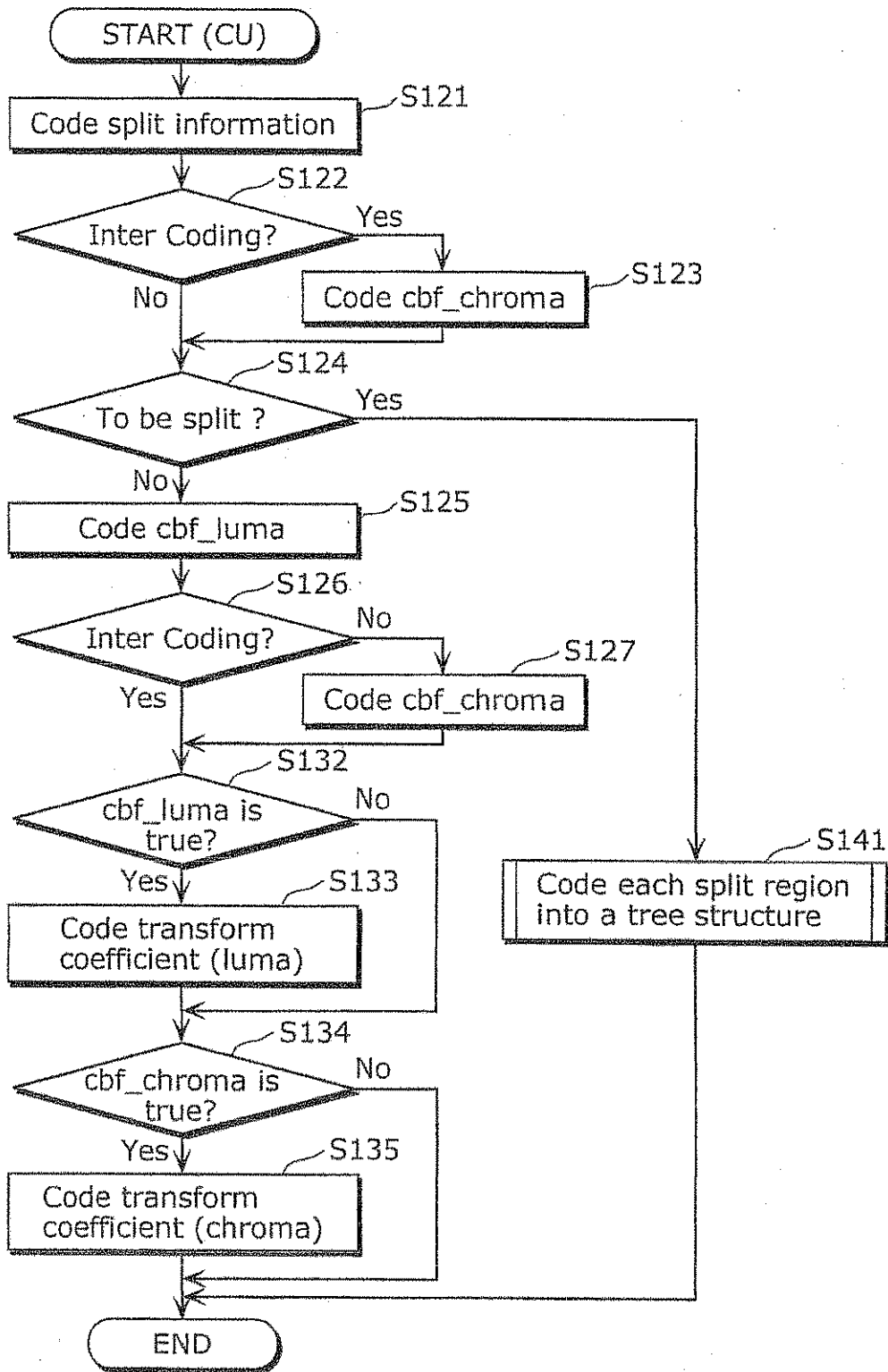


FIG. 16

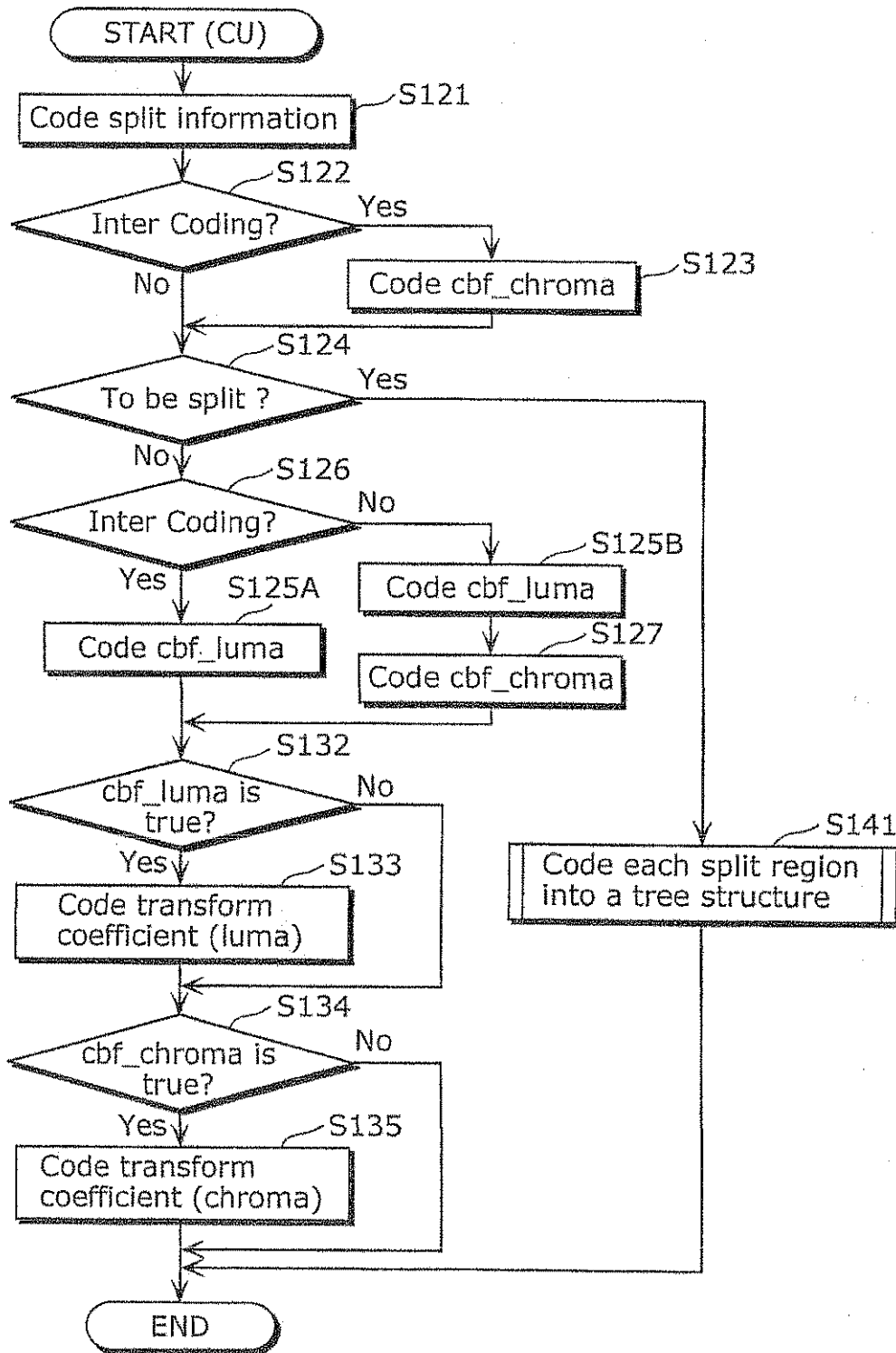


FIG. 17

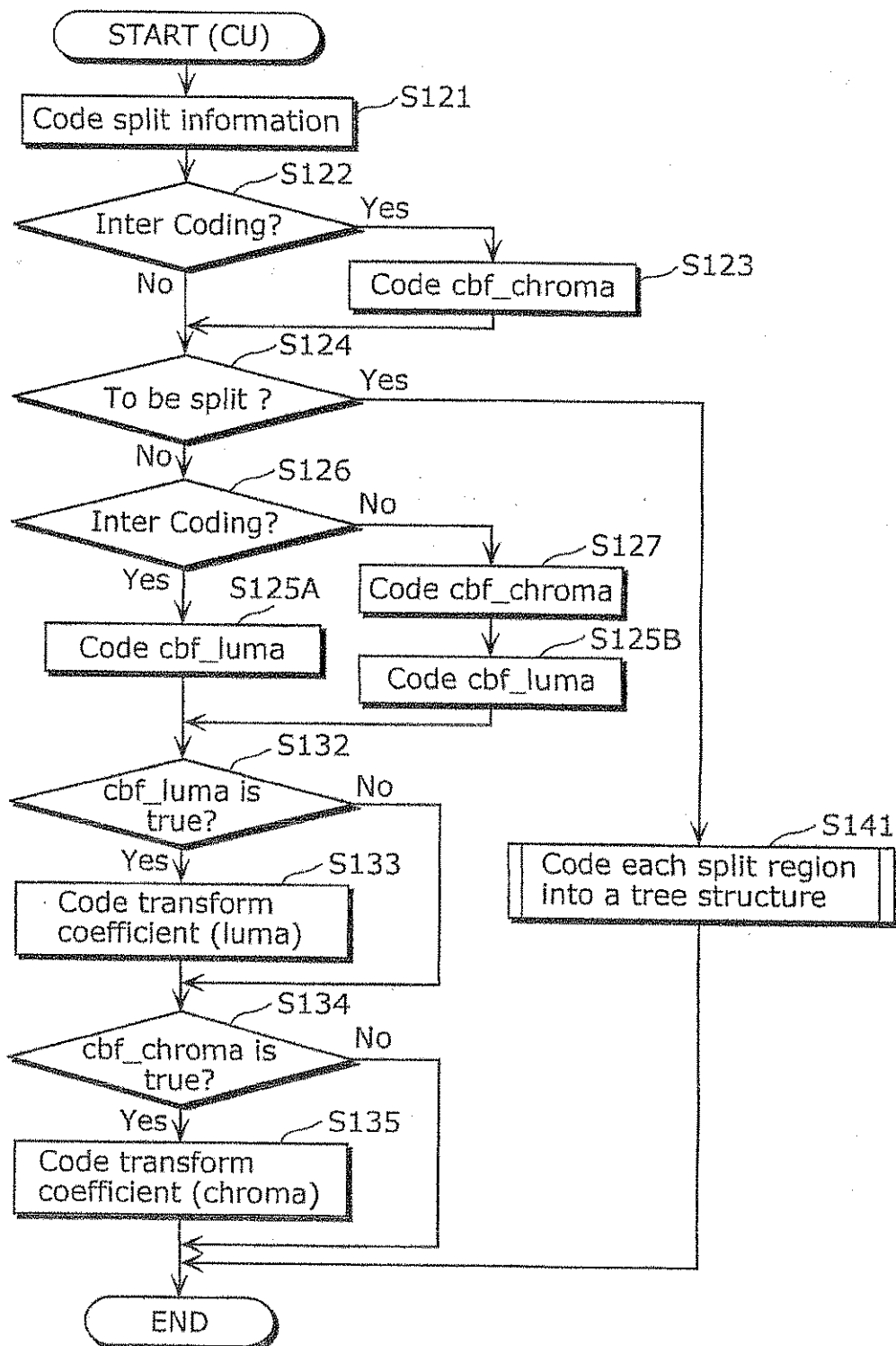


FIG. 18A

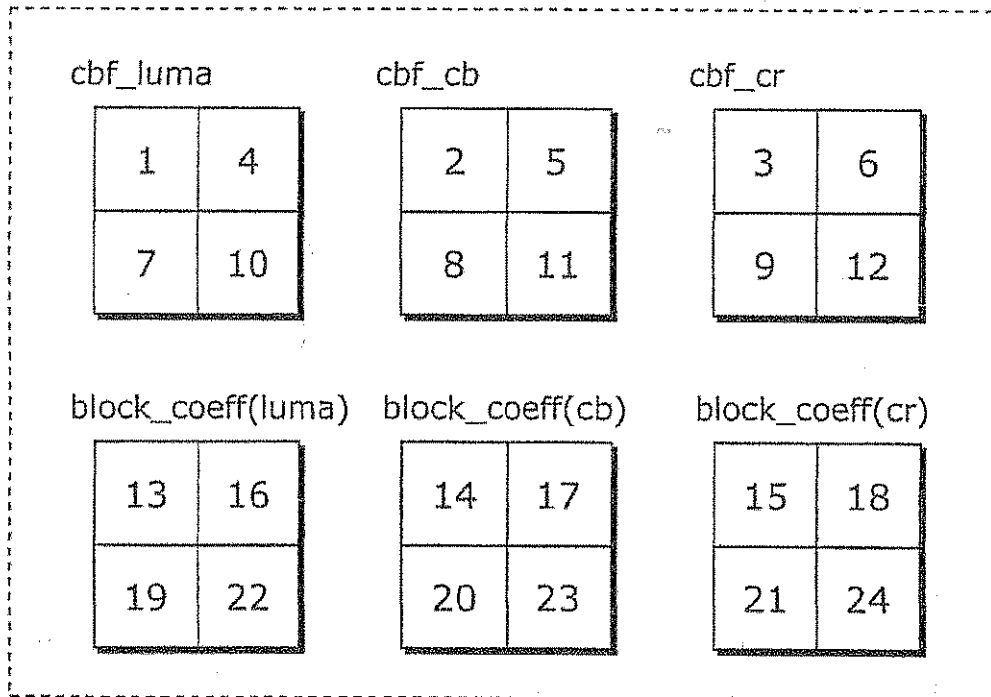


FIG. 18B

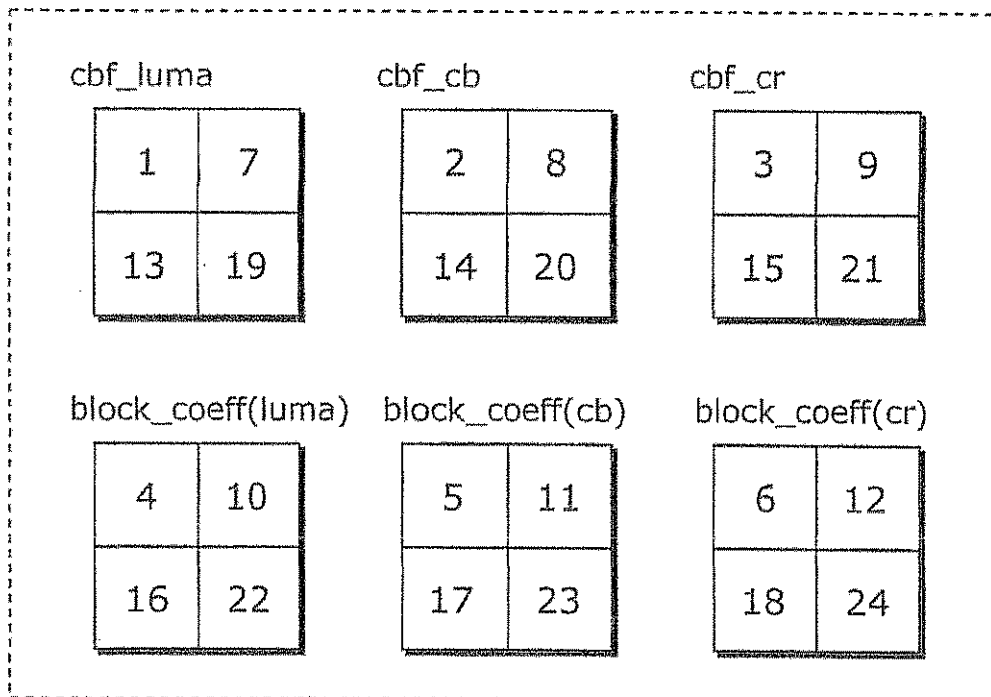


FIG. 18C

cbf_luma	cbf_cb	cbf_cr												
<table><tr><td>1</td><td>7</td></tr><tr><td>13</td><td>19</td></tr></table>	1	7	13	19	<table><tr><td>3</td><td>9</td></tr><tr><td>15</td><td>21</td></tr></table>	3	9	15	21	<table><tr><td>5</td><td>11</td></tr><tr><td>17</td><td>23</td></tr></table>	5	11	17	23
1	7													
13	19													
3	9													
15	21													
5	11													
17	23													
block_coeff(luma)	block_coeff(cb)	block_coeff(cr)												
<table><tr><td>2</td><td>8</td></tr><tr><td>14</td><td>20</td></tr></table>	2	8	14	20	<table><tr><td>4</td><td>10</td></tr><tr><td>16</td><td>22</td></tr></table>	4	10	16	22	<table><tr><td>6</td><td>12</td></tr><tr><td>18</td><td>24</td></tr></table>	6	12	18	24
2	8													
14	20													
4	10													
16	22													
6	12													
18	24													

FIG. 19A

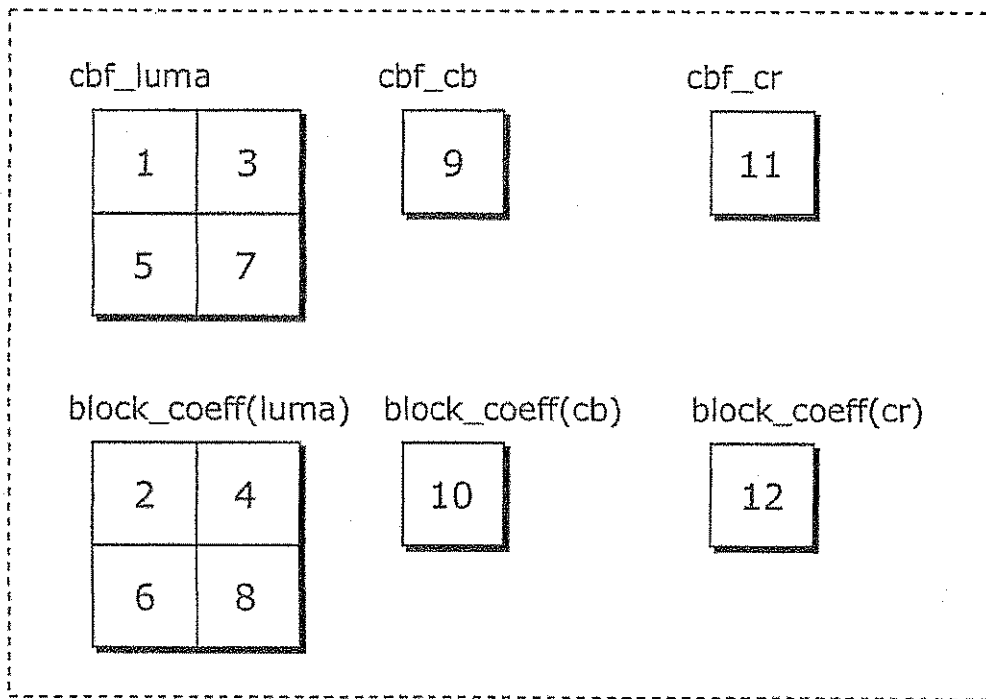


FIG. 19B

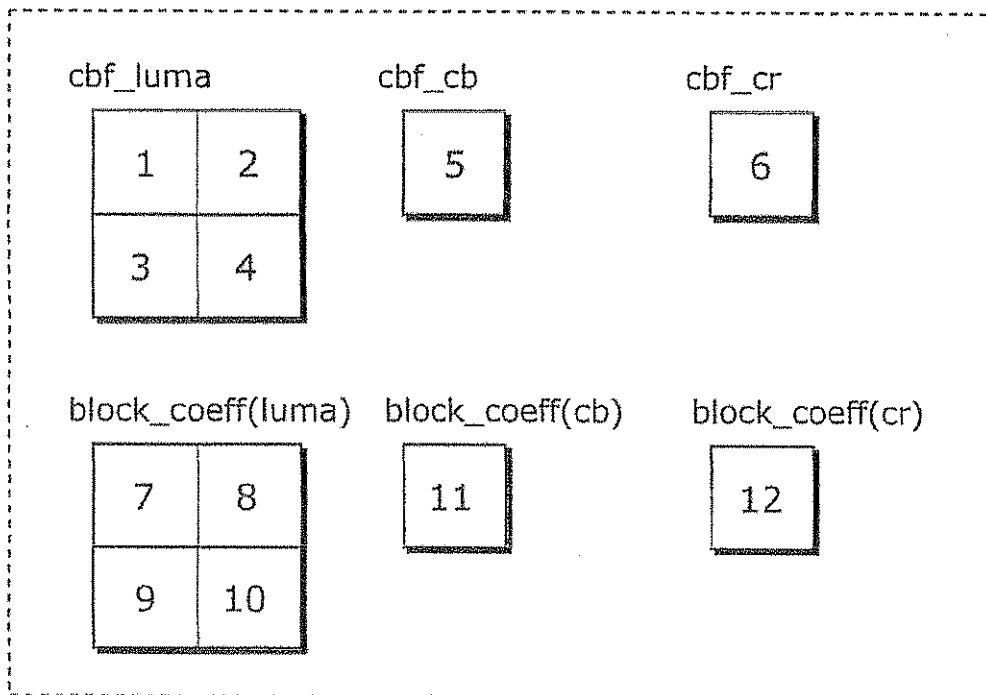


FIG. 20

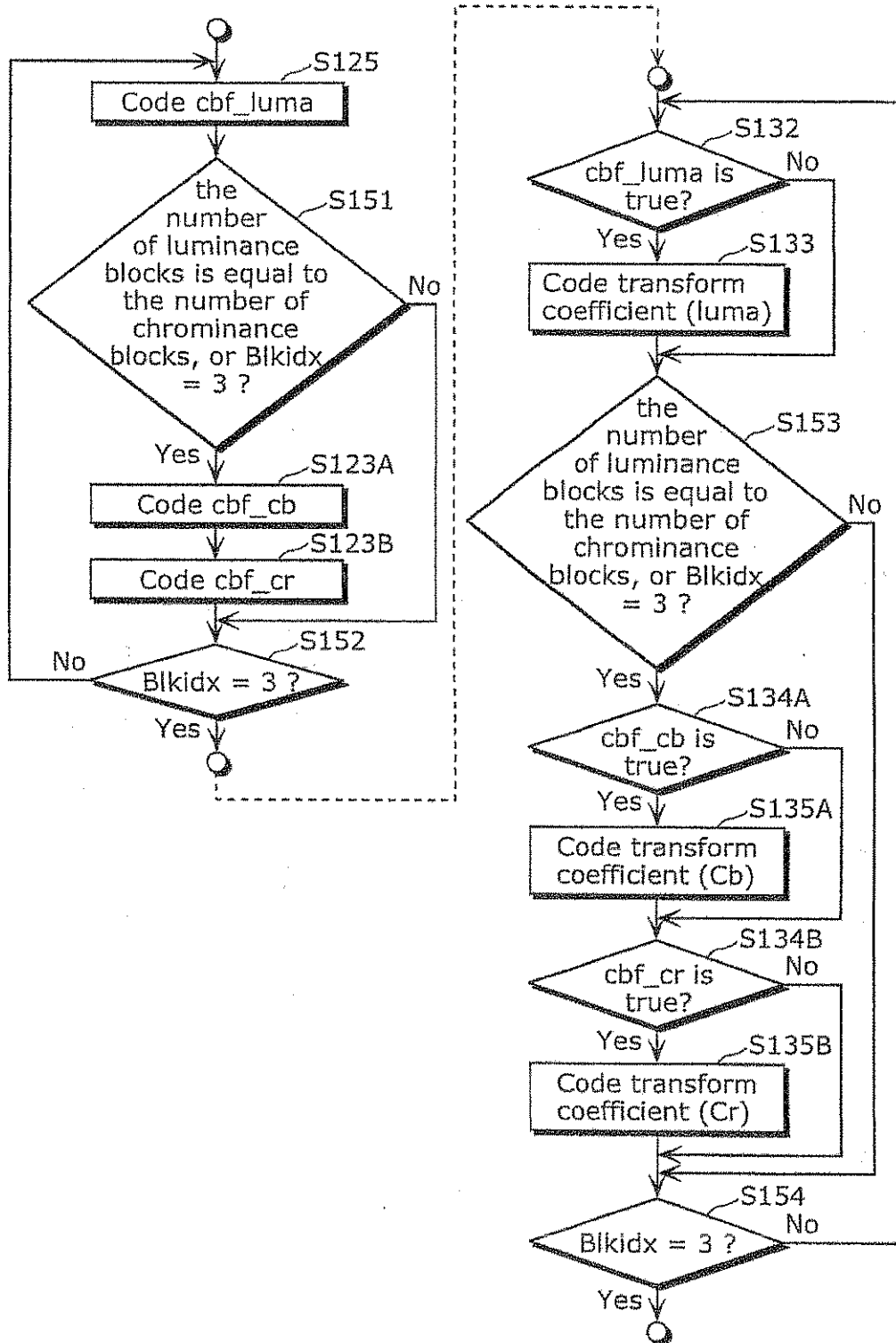


FIG. 21A

cbf_luma	cbf_cb	cbf_cr												
<table><tr><td>3</td><td>9</td></tr><tr><td>15</td><td>21</td></tr></table>	3	9	15	21	<table><tr><td>1</td><td>7</td></tr><tr><td>13</td><td>19</td></tr></table>	1	7	13	19	<table><tr><td>2</td><td>8</td></tr><tr><td>14</td><td>20</td></tr></table>	2	8	14	20
3	9													
15	21													
1	7													
13	19													
2	8													
14	20													
block_coeff(luma)	block_coeff(cb)	block_coeff(cr)												
<table><tr><td>4</td><td>10</td></tr><tr><td>16</td><td>22</td></tr></table>	4	10	16	22	<table><tr><td>5</td><td>11</td></tr><tr><td>17</td><td>23</td></tr></table>	5	11	17	23	<table><tr><td>6</td><td>12</td></tr><tr><td>18</td><td>24</td></tr></table>	6	12	18	24
4	10													
16	22													
5	11													
17	23													
6	12													
18	24													

FIG. 21B

cbf_luma	cbf_cb	cbf_cr						
<table><tr><td>3</td><td>7</td></tr><tr><td>9</td><td>11</td></tr></table>	3	7	9	11	<table><tr><td>1</td></tr></table>	1	<table><tr><td>2</td></tr></table>	2
3	7							
9	11							
1								
2								
block_coeff(luma)	block_coeff(cb)	block_coeff(cr)						
<table><tr><td>4</td><td>8</td></tr><tr><td>10</td><td>12</td></tr></table>	4	8	10	12	<table><tr><td>5</td></tr></table>	5	<table><tr><td>6</td></tr></table>	6
4	8							
10	12							
5								
6								

FIG. 22A

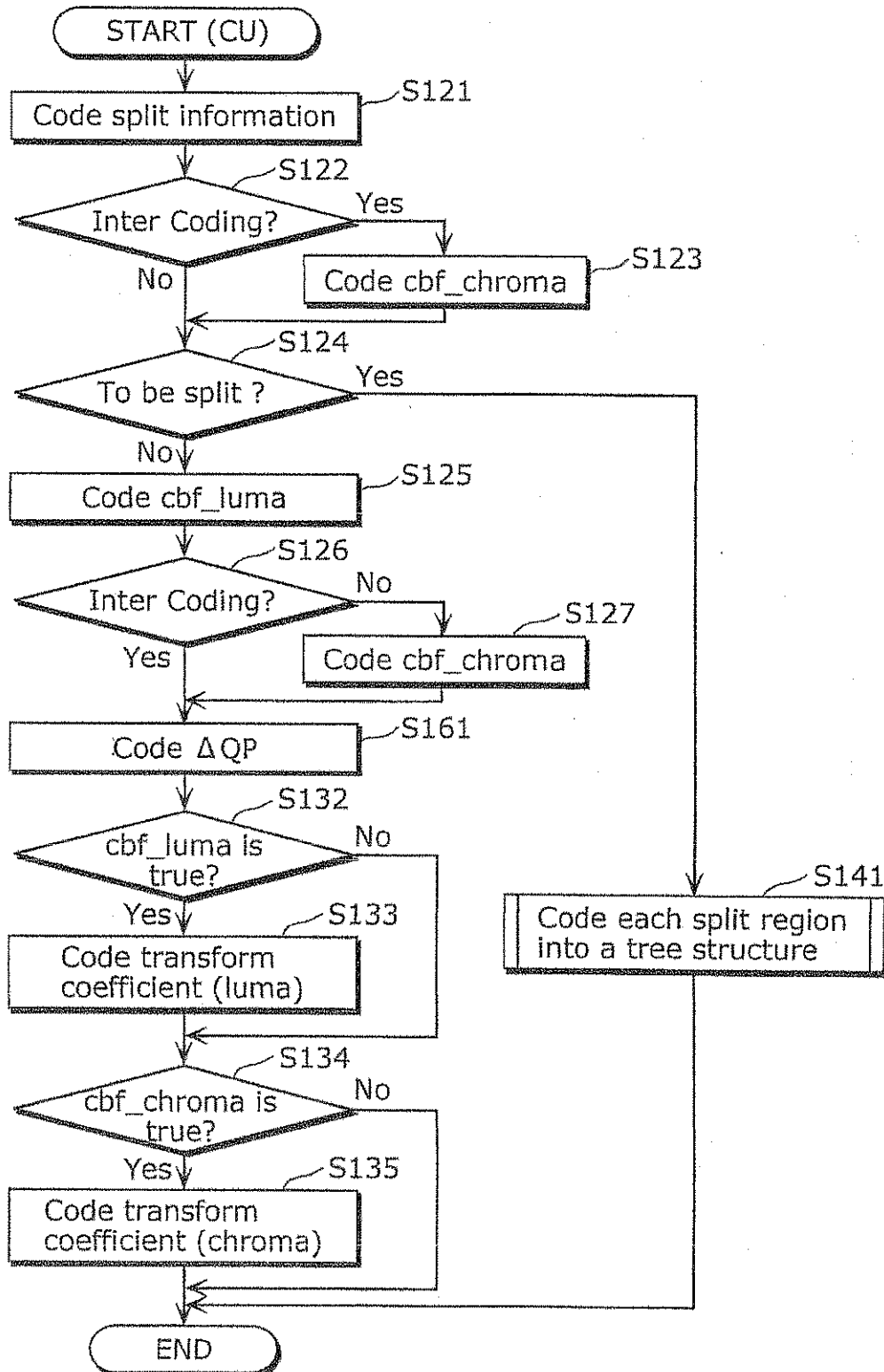


FIG. 22B

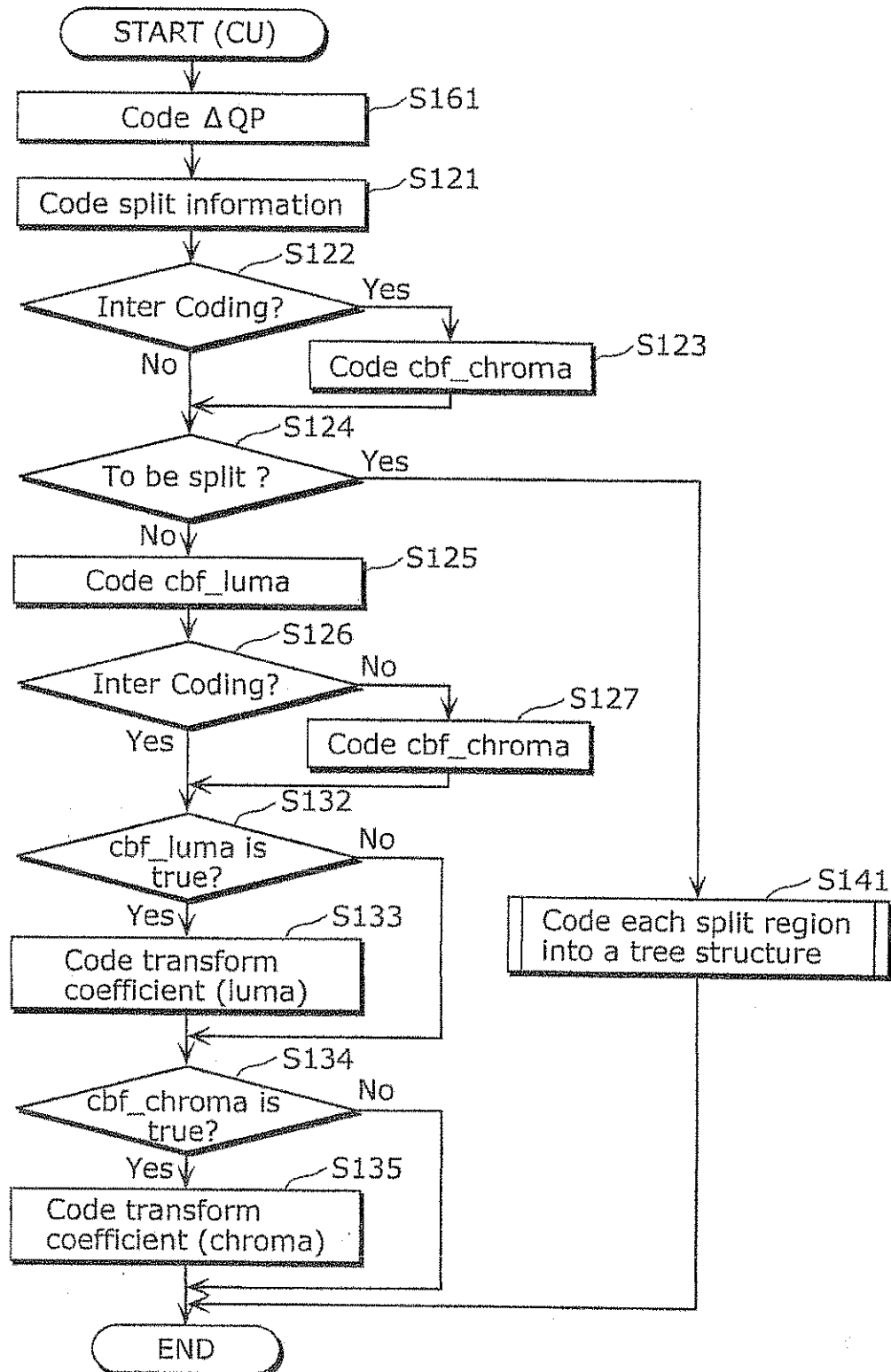


FIG. 23

```

transform_tree( x0,y0, log2TrafoWidth, log2TrafoHeight, trafoDepth, blkIdx )
{
    split_transform_flag[ x0 ][ y0 ][ trafoDepth ]
    if( PredMode == Inter)
    {
        cbf_cb[ x0 ][ y0 ][ trafoDepth ]
        cbf_cr[ x0 ][ y0 ][ trafoDepth ]
    }
    if( split_transform_flag[ x0 ][ y0 ][ trafoDepth ] )
    {
        // splitting into 4 regions and recursive call
    }
    else
    {
        if( PredMode == MODE_INTER)
        {
            cbf_luma[ x0 ][ y0 ][ trafoDepth ]
        }
        if( PredMode == MODE_INTRA )
        {
            cbf_luma[ x0 ][ y0 ][ trafoDepth ]
            cbf_cb[ x0 ][ y0 ][ trafoDepth ]
            cbf_cr[ x0 ][ y0 ][ trafoDepth ]
        }
        cu_qp_delta
        if( cbf_luma[ x0 ][ y0 ][ trafoDepth ] )
            residual_coding_cabac( x0, y0, log2TrafoSize, trafoDepth, scanIdx, cIdx 0 )
        if( cbf_cb[ x0 ][ y0 ][ trafoDepth ] )
            residual_coding_cabac( x0, y0, log2TrafoSize, trafoDepth, scanIdxC, 1 )
        if( cbf_cr[ x0 ][ y0 ][ trafoDepth ] )
            residual_coding_cabac( x0, y0, log2TrafoSize, trafoDepth, scanIdxC, 2 )
    }
}

```

FIG. 24A

```

transform_tree( x0,y0, log2TrafoWidth, log2TrafoHeight, trafoDepth, blkIdx){
  if( trafoDepth == 0 && IntraSplitFlag == 0) // IntraSplitFlag: 2Nx2N -> 0, NxN ->1
  {
    if( PredMode != MODE_INTRA )
      no_residual_data_flag
      residualDataPresentFlag = !no_residual_data_flag
    } else {
      residualDataPresentFlag = TRUE
    }
    if( residualDataPresentFlag)
    {
      log2TrafoSize = ( log2TrafoWidth + log2TrafoHeight ) >> 1
      intraSplitFlag = ( IntraSplitFlag && trafoDepth == 0 ? 1 : 0 )
      maxDepth = ( PredMode == MODE_INTRA ?
        max_transform_hierarchy_depth_intra + IntraSplitFlag :
        max_transform_hierarchy_depth_inter )
      xBase = x0 - ( x0 & ( 1 << log2TrafoWidth ) )
      yBase = y0 - ( y0 & ( 1 << log2TrafoHeight ) )
      if( log2TrafoSize <= Log2MaxTrafoSize &&
        log2TrafoSize > Log2MinTrafoSize &&
        trafoDepth < maxDepth && !intraSplitFlag && entropy_coding_mode_flag )
        split_transform_flag[ x0 ] [ y0 ] [ trafoDepth ]
      if( PredMode != MODE_INTRA &&
        log2TrafoSize <= Log2MaxTrafoSize &&
        entropy_coding_mode_flag ){
        firstChromaCbf = ( log2TrafoSize == Log2MaxTrafoSize ||
          trafoDepth == 0 ? 1 : 0 )
        if( firstChromaCbf || log2TrafoSize > Log2MinTrafoSize ){
          if( firstChromaCbf || cbf_cb[ xBase ] [ yBase ] [ trafoDepth - 1 ] ){
            readCbf = true
            if( blkIdx == 3 && log2TrafoSize < Log2MaxTrafoSize )
              readCbf = cbf_cb[ xBase ] [ yBase ] [ trafoDepth ] ||
                cbf_cb[ xBase + ( 1 << log2TrafoWidth ) ] [ yBase ] [ trafoDepth ] ||
                cbf_cb[ xBase ] [ yBase + ( 1 << log2TrafoHeight ) ] [ trafoDepth ]
            if( !readCbf )
              cbf_cb[ x0 ] [ y0 ] [ trafoDepth ] = 1
            else
              cbf_cb[ x0 ] [ y0 ] [ trafoDepth ]
          }
          if( firstChromaCbf || cbf_cr[ xBase ] [ yBase ] [ trafoDepth ? 1 ] ){
            readCbf = true
            if( blkIdx == 3 && log2TrafoSize < Log2MaxTrafoSize )
              readCbf = cbf_cr[ xBase ] [ yBase ] [ trafoDepth ] ||
                cbf_cr[ xBase + ( 1 << log2TrafoWidth ) ] [ yBase ] [ trafoDepth ] ||
                cbf_cr[ xBase ] [ yBase + ( 1 << log2TrafoHeight ) ] [ trafoDepth ]
            if( !readCbf )
              cbf_cr[ x0 ] [ y0 ] [ trafoDepth ] = 1
            else
              cbf_cr[ x0 ] [ y0 ] [ trafoDepth ]
          }
        }
      }
    }
  }
}

```

FIG. 24B

```

:
:
if( split_transform_flag[ x0 ][ y0 ][ trafoDepth ] ){
    if( InterTUSplitDirection == 2 ){ // square split
        x1 = x0 + (( 1 << log2TrafoWidth ) >> 1 )
        y1 = y0
        x2 = x0
        y2 = y0 + (( 1 << log2TrafoHeight ) >> 1 )
        x3 = x1
        y3 = y2
    } else {
        x1 = x0 + ((1 << log2TrafoWidth ) >> 2 ) * InterTUSplitDirection
        y1 = y0 + ((1 << log2TrafoHeight ) >> 2 ) * (1 - InterTUSplitDirection )
        x2 = x1 + ((1 << log2TrafoWidth ) >> 2 ) * InterTUSplitDirection
        y2 = y1 + ((1 << log2TrafoHeight ) >> 2 ) * (1 - InterTUSplitDirection )
        x3 = x2 + ((1 << log2TrafoWidth ) >> 2 ) * InterTUSplitDirection
        y3 = y2 + ((1 << log2TrafoHeight ) >> 2 ) * (1 - InterTUSplitDirection )
        log2TrafoHeight = log2TrafoHeight + 2 * InterTUSplitDirection - 1
        log2TrafoWidth = log2TrafoWidth - 2 * InterTUSplitDirection + 1
    }
    transform_tree(x0, y0, x0, y0, log2TrafoWidth - 1, log2TrafoHeight -1, trafoDepth + 1, 0 )
    transform_tree(x1, y1, x0, y0, log2TrafoWidth - 1, log2TrafoHeight -1, trafoDepth + 1, 1 )
    transform_tree(x2, y2, x0, y0, log2TrafoWidth - 1, log2TrafoHeight -1, trafoDepth + 1, 2 )
    transform_tree(x3, y3, x0, y0, log2TrafoWidth - 1, log2TrafoHeight -1, trafoDepth + 1, 3 )
} else{
    if( PredMode != MODE_INTRA && ( trafoDepth != 0 ||
        cbf_cb[ x0 ][ y0 ][ trafoDepth ] ||
        cbf_cr[ x0 ][ y0 ][ trafoDepth ] ) ){
        readCbf = true
        if( blkIdx == 3 && log2TrafoSize < Log2MaxTrafoSize )
            readCbf = cbf_luma[ xBase ][ yBase ][ trafoDepth ] ||
            cbf_luma[ xBase + (1 << log2TrafoWidth ) ][ yBase ][ trafoDepth ] ||
            cbf_luma[ xBase ][ yBase + (1 << log2TrafoHeight ) ][ trafoDepth ] ||
            cbf_cb[ xBase ][ yBase ][ trafoDepth - 1 ] ||
            cbf_cr[ xBase ][ yBase ][ trafoDepth - 1 ]
        if ( !readCbf )
            cbf_luma[ x0 ][ y0 ][ trafoDepth ] = 1
        else
            cbf_luma[ x0 ][ y0 ][ trafoDepth ]
    }
    if( PredMode == MODE_INTRA )
    {
        cbf_luma[ x0 ][ y0 ][ trafoDepth ]
        log2TrafoSize = ( log2TrafoWidth + log2TrafoHeight ) >> 1
        log2TrafoSizeC = ( log2TrafoSize == Log2MinTrafoSize ) ? log2TrafoSize : (log2TrafoSize - 1)
        if((log2TrafoSize > Log2MinTrafoSize) || (log2TrafoSize == Log2MinTrafoSize && blk == 3 ))
        {
            cbf_cb[ x0 ][ y0 ][ trafoDepth ]
            cbf_cr[ x0 ][ y0 ][ trafoDepth ]
        }
    }
}
:
:

```


FIG. 25A

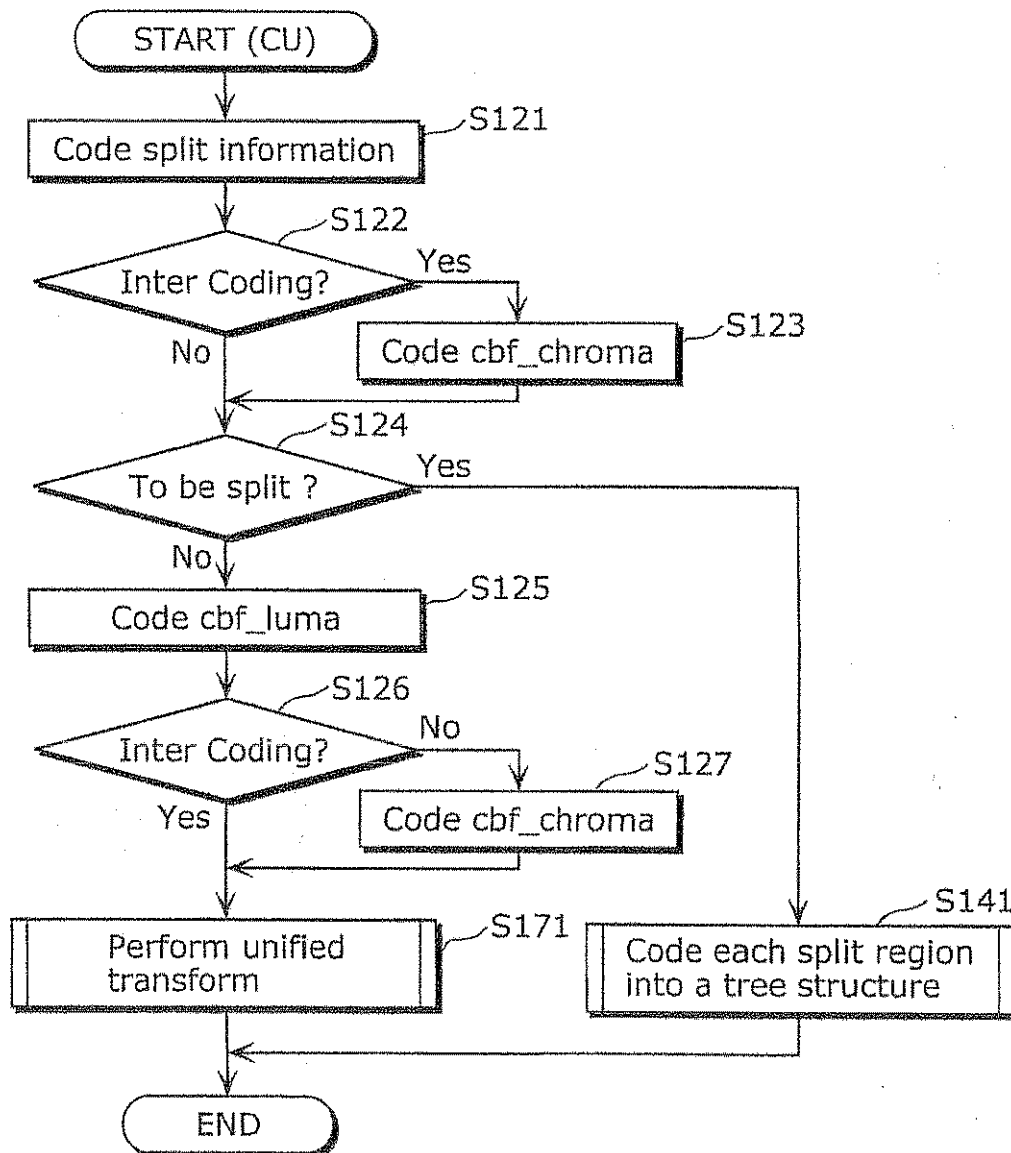


FIG. 25B

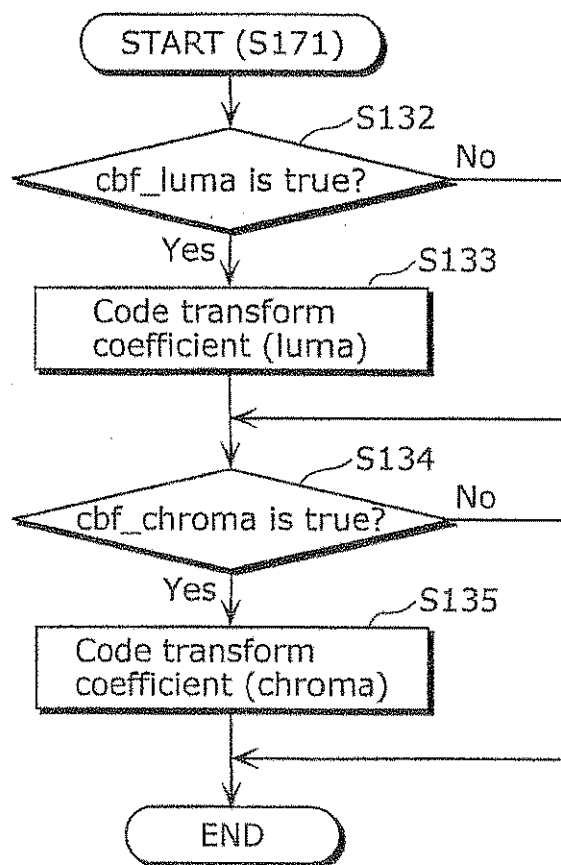


FIG. 26

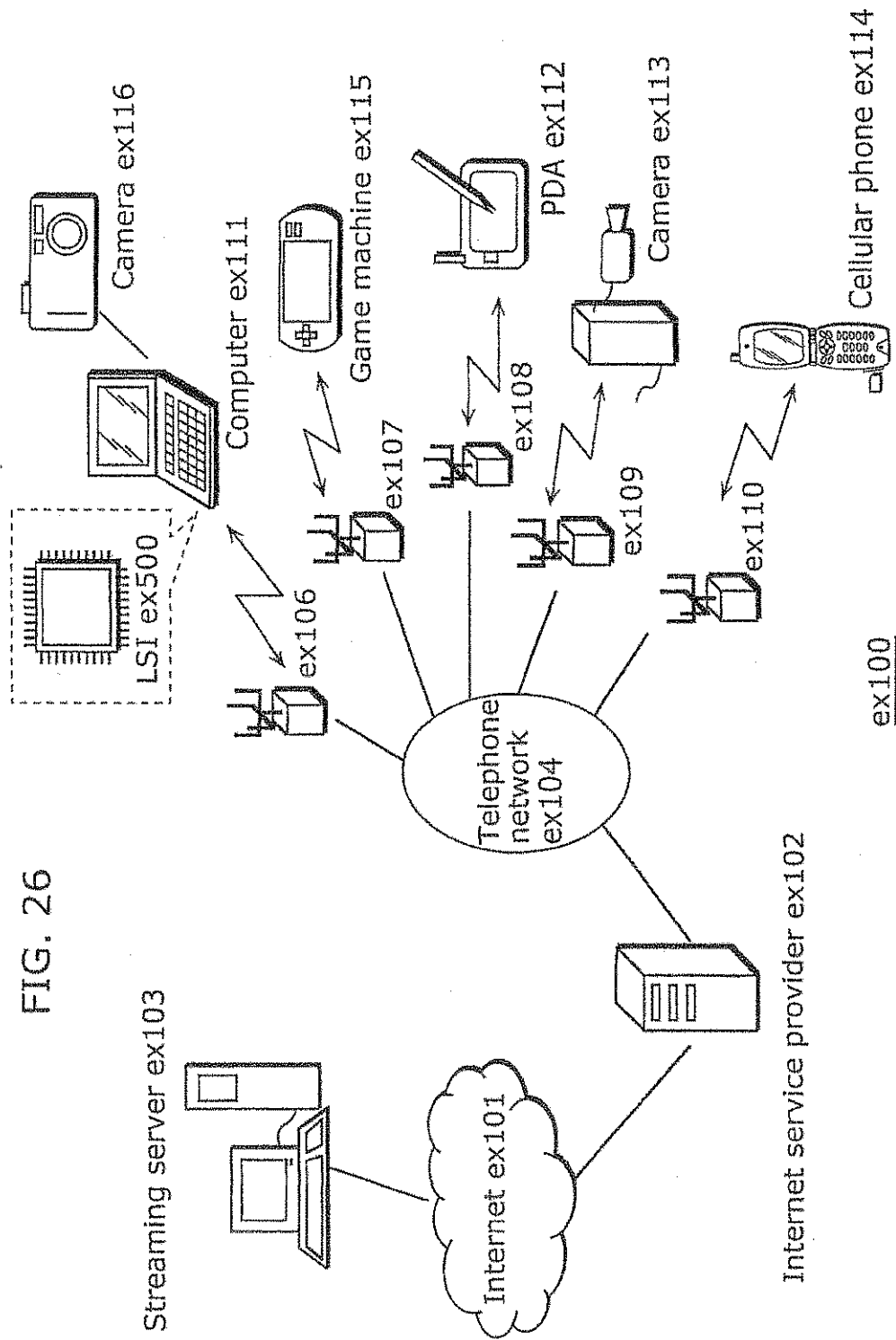


FIG. 27

Diagram illustrating a broadcast system (FIG. 27) showing a Broadcast station ex201, a Broadcast satellite ex202, and various receiving devices connected via Cable ex203 and Antenna ex204.

The system includes the following components:

- Broadcast station ex201
- Broadcast satellite ex202
- Antenna ex204
- Car ex210
- Car navigation system ex211
- Cellular phone ex114
- Reproduction apparatus ex212
- Monitor ex213
- Monitor ex214
- Antenna ex205
- Car ex210
- Car navigation system ex211
- Antenna ex204
- Broadcast station ex201
- Cable ex203
- STB ex217
- Monitor ex219
- Reader/recorder ex218
- Television ex300
- ex200
- ex215
- ex216

FIG. 28

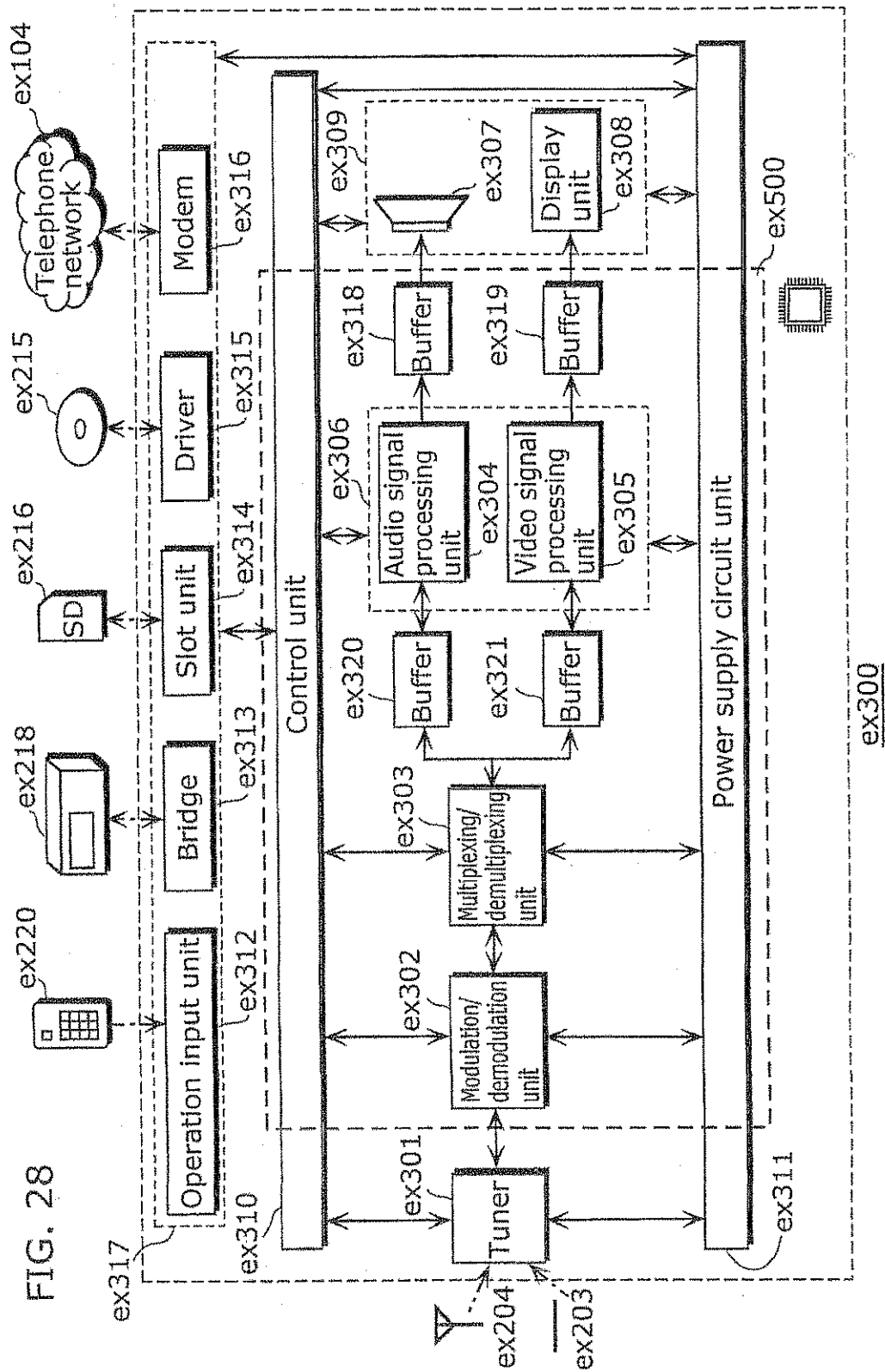
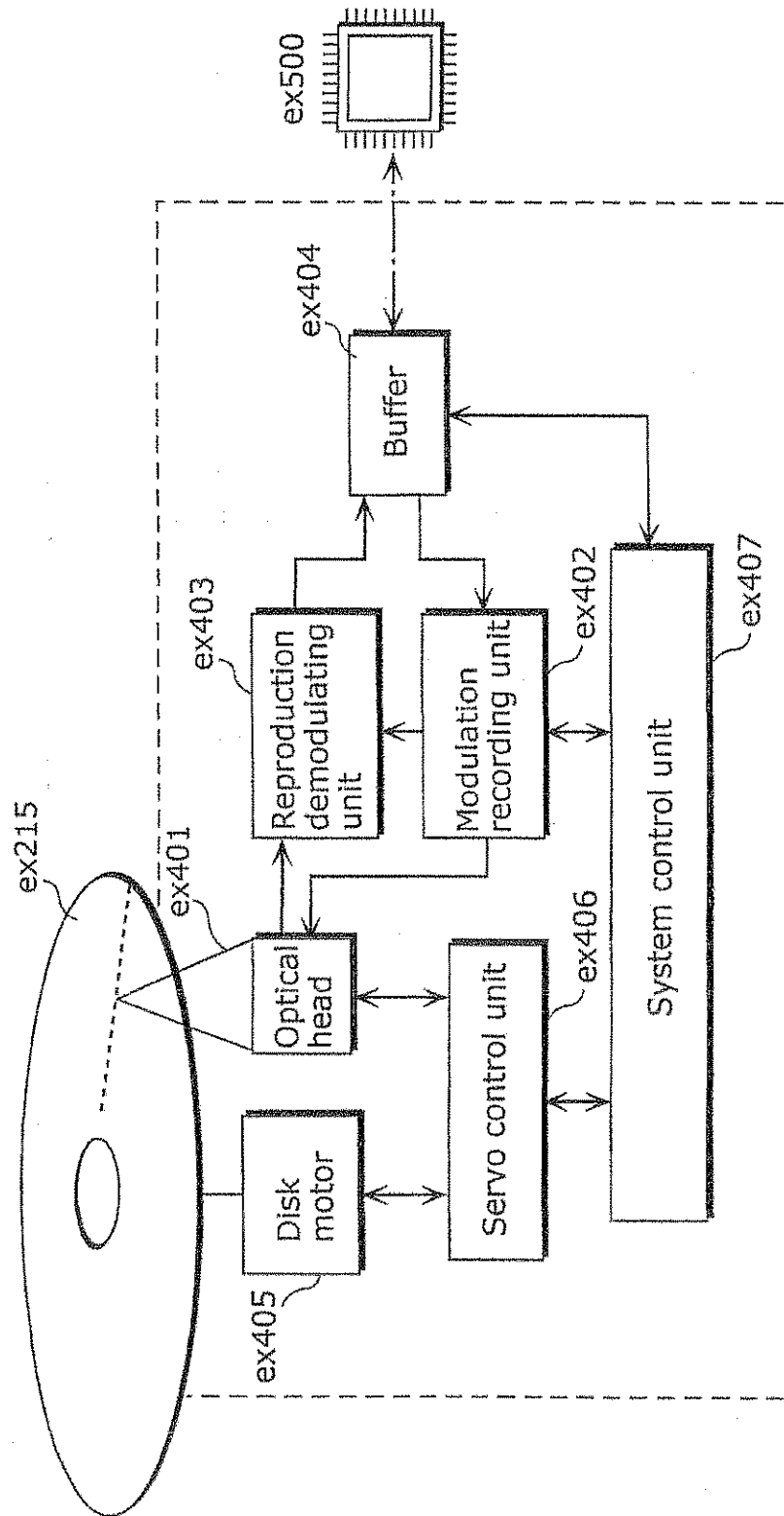


FIG. 29



ex400

FIG. 30

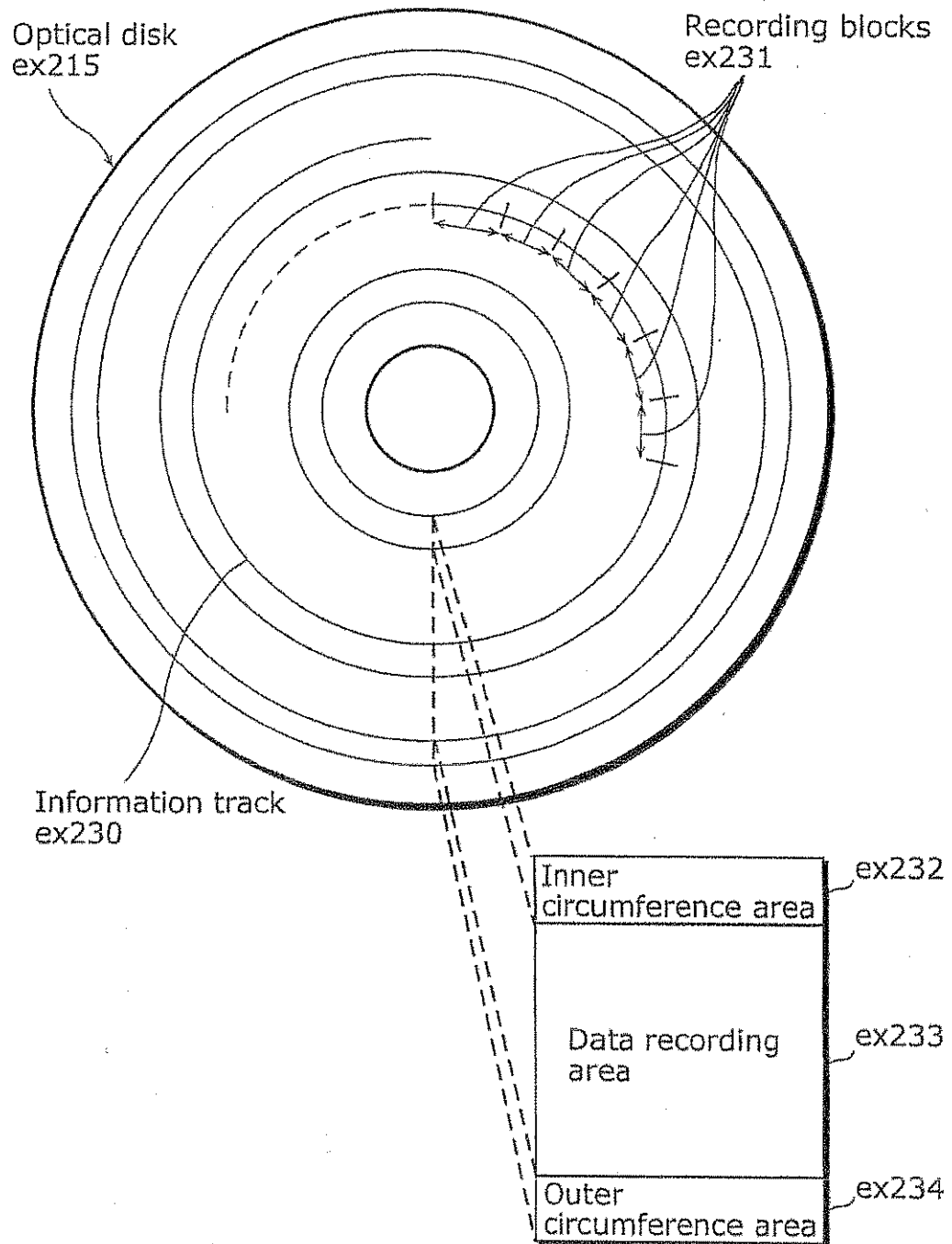


FIG. 31A

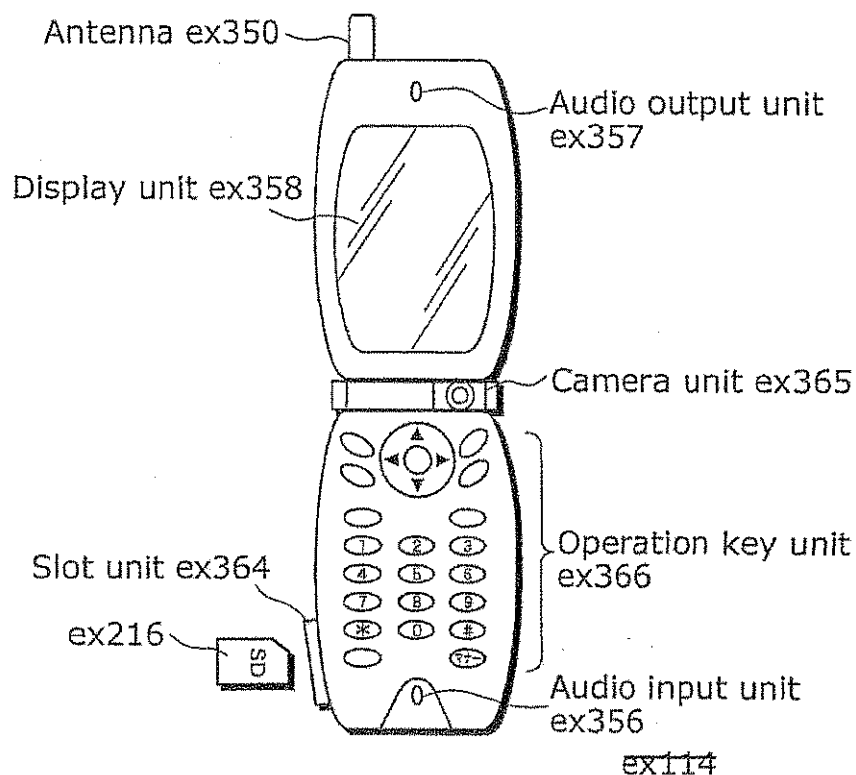


FIG. 31B

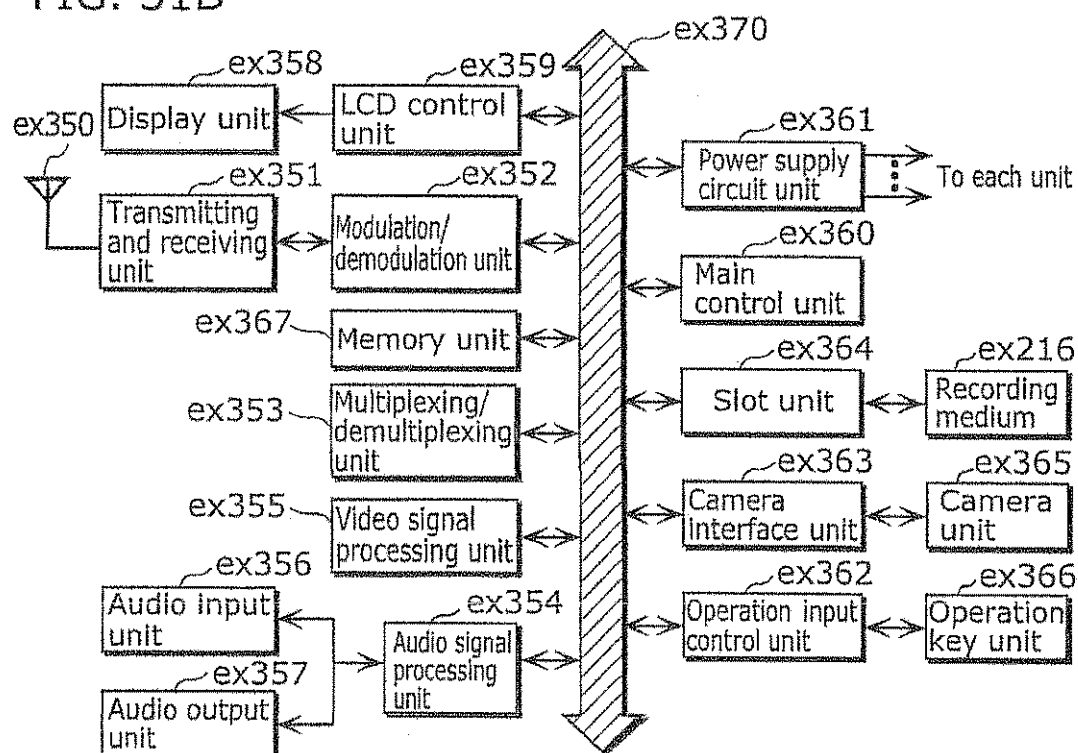


FIG. 32

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. 33

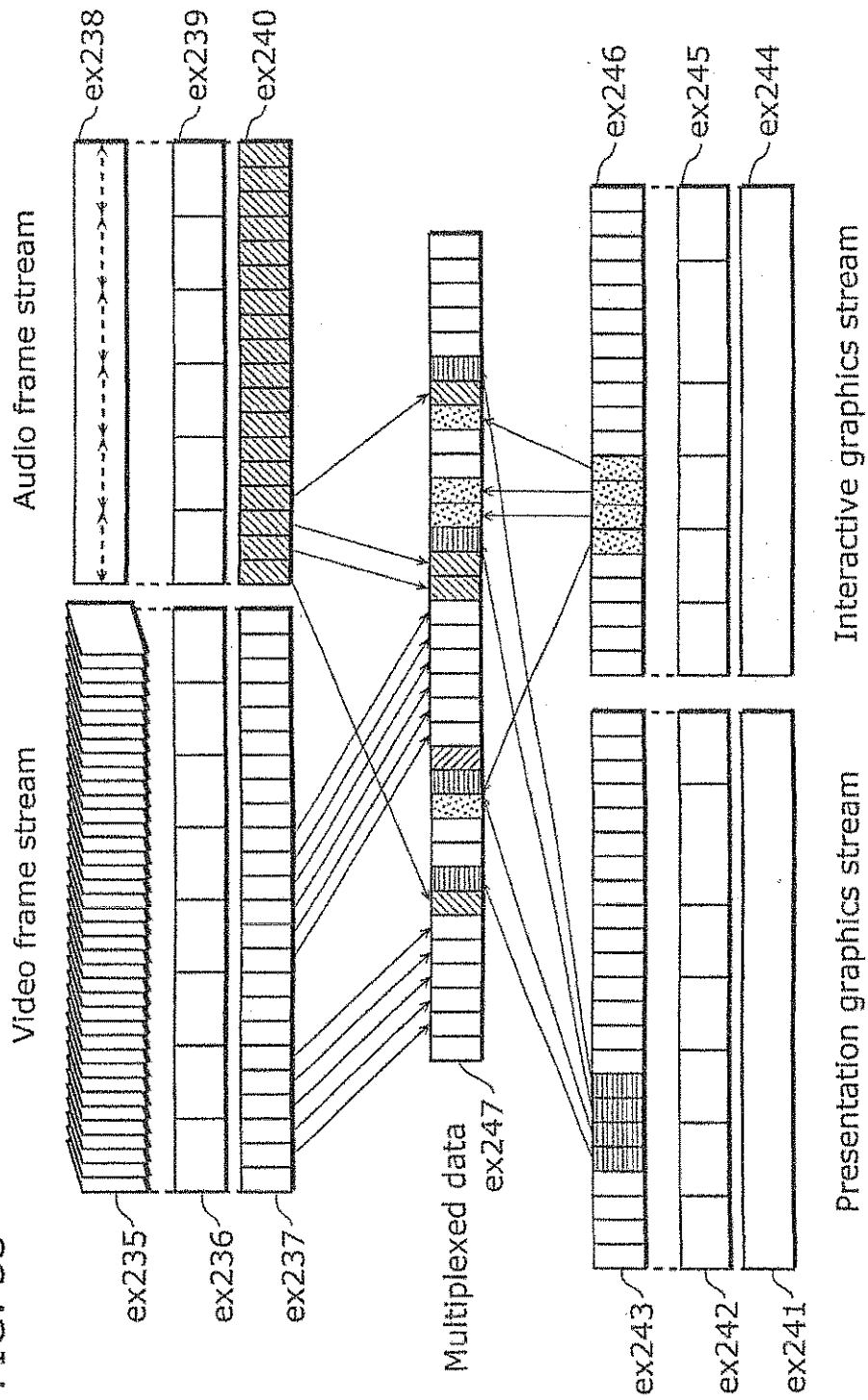


FIG. 34

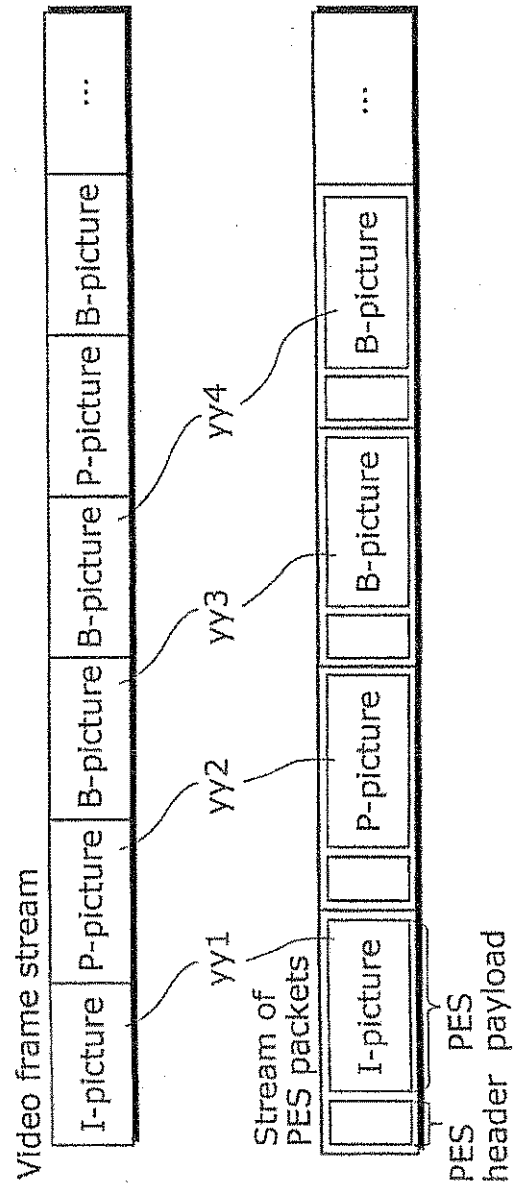


FIG. 35

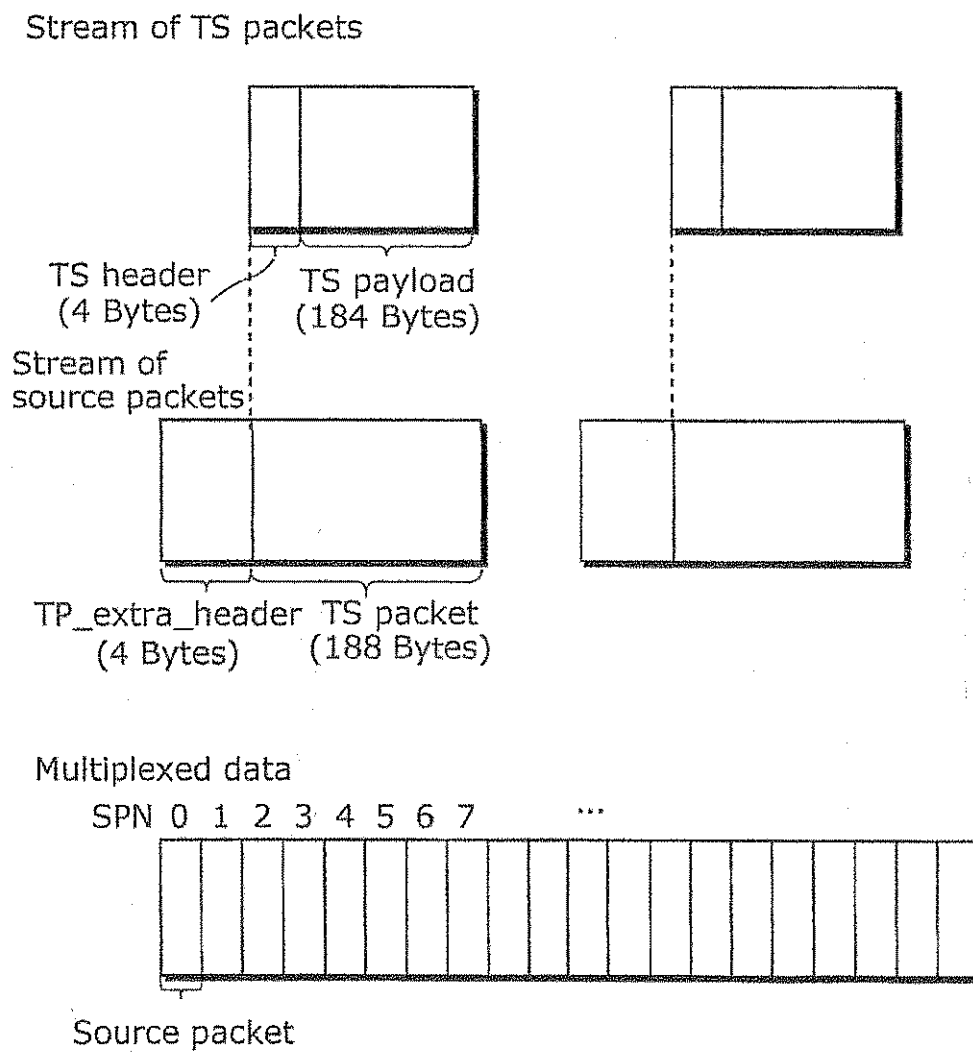


FIG. 36

Data structure of PMT

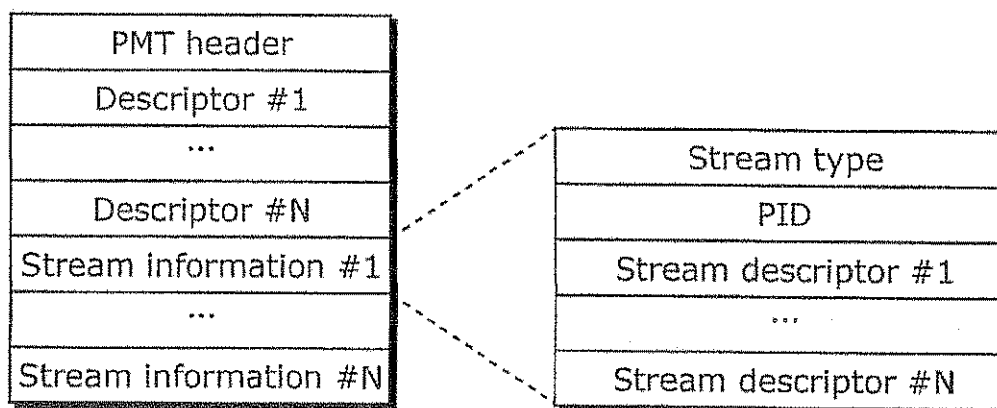


FIG. 37

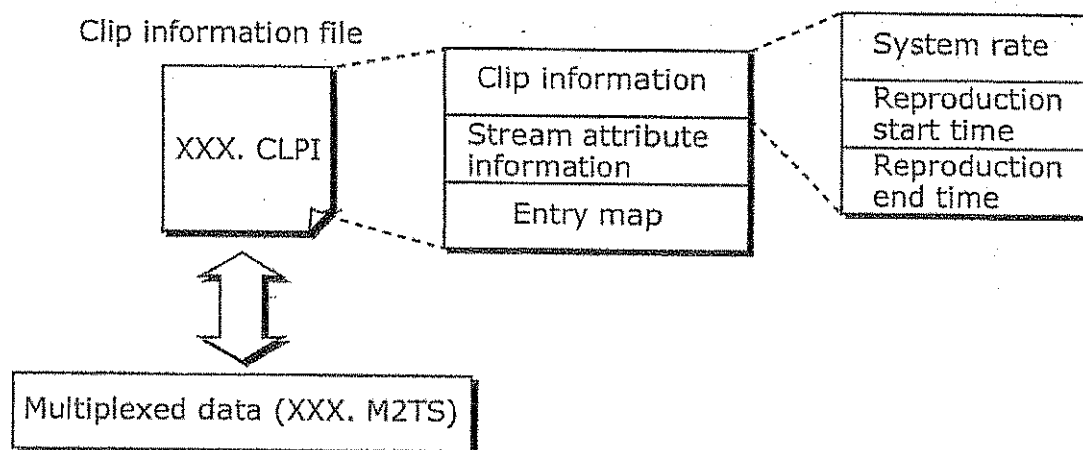


FIG. 38

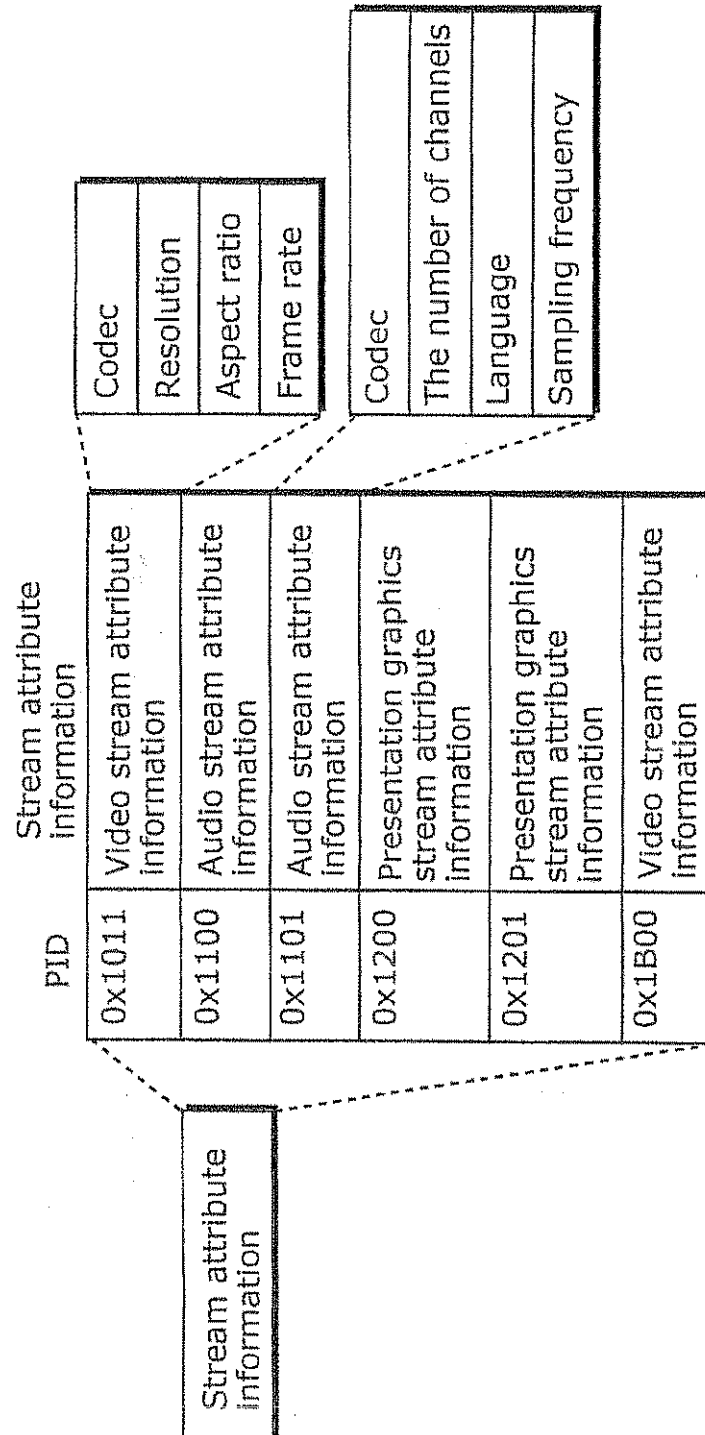


FIG. 39

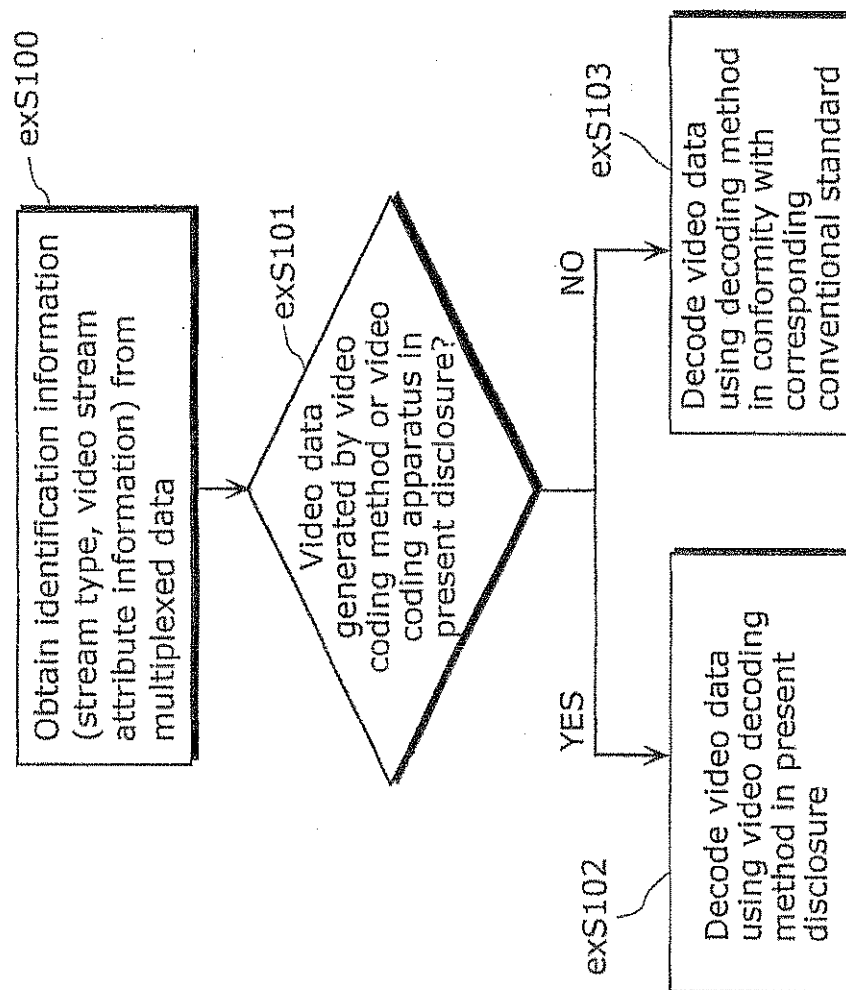


FIG. 40

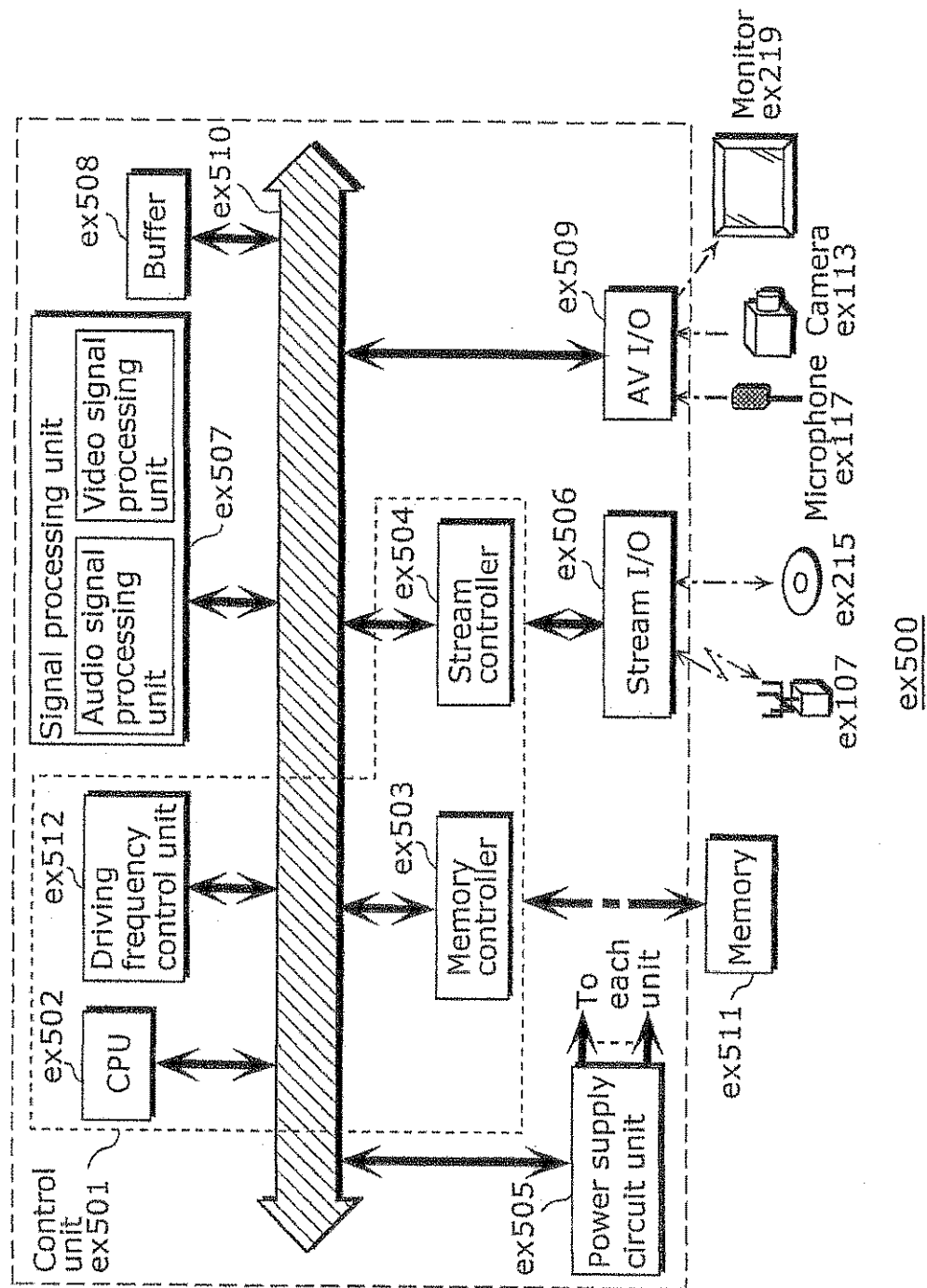


FIG. 41

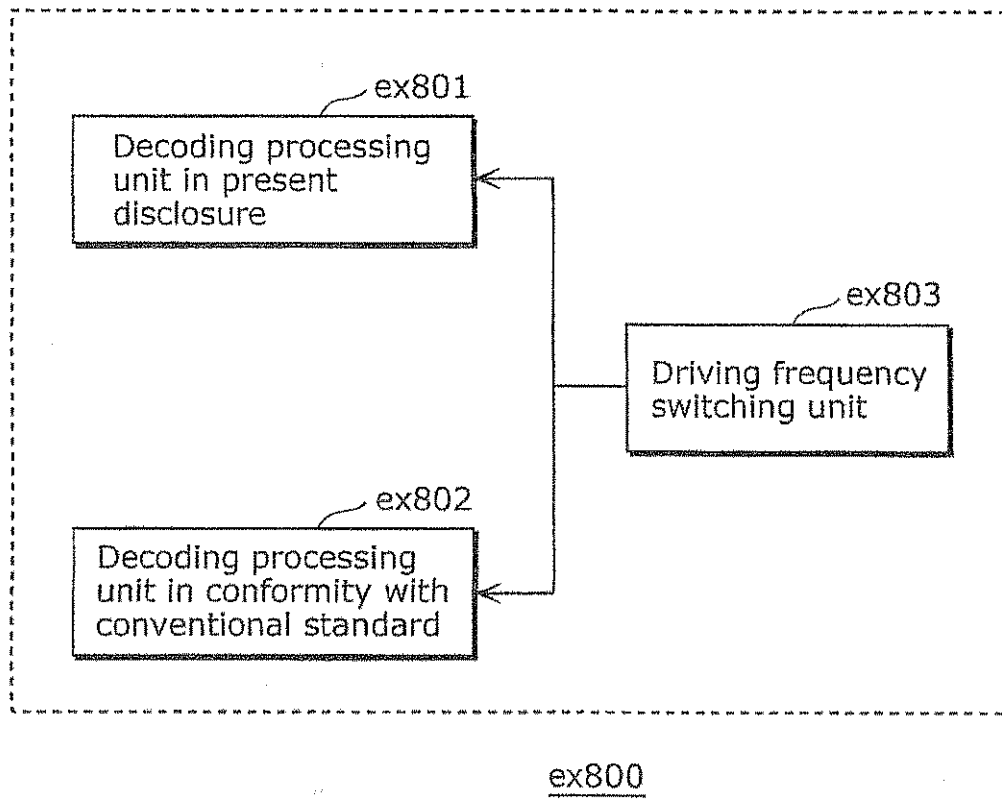


FIG. 42

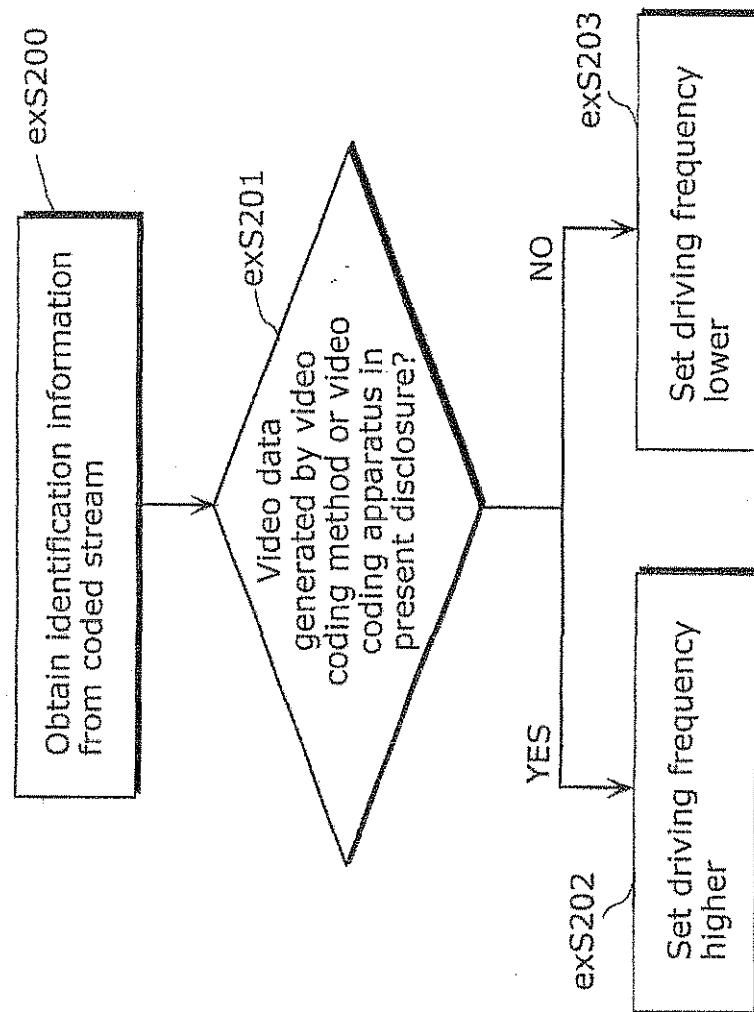


FIG. 43

Corresponding standard	Driving frequency
MPEG4.AVC	500 MHz
MPEG-2	350 MHz
⋮	⋮

FIG. 44A

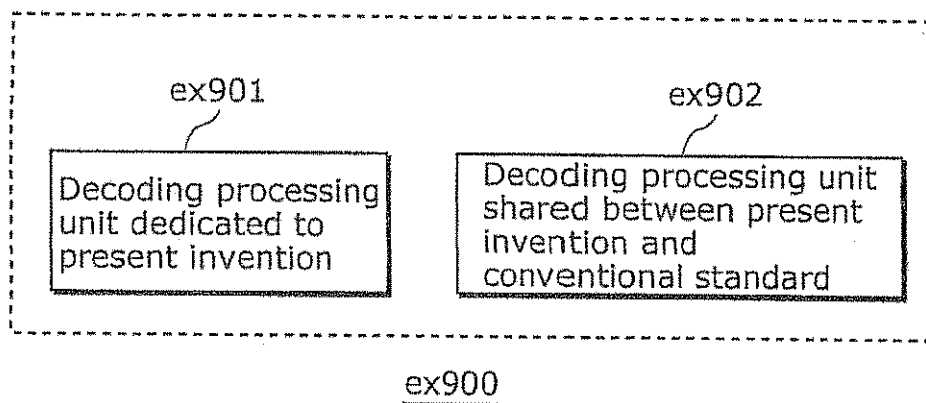


FIG. 44B

