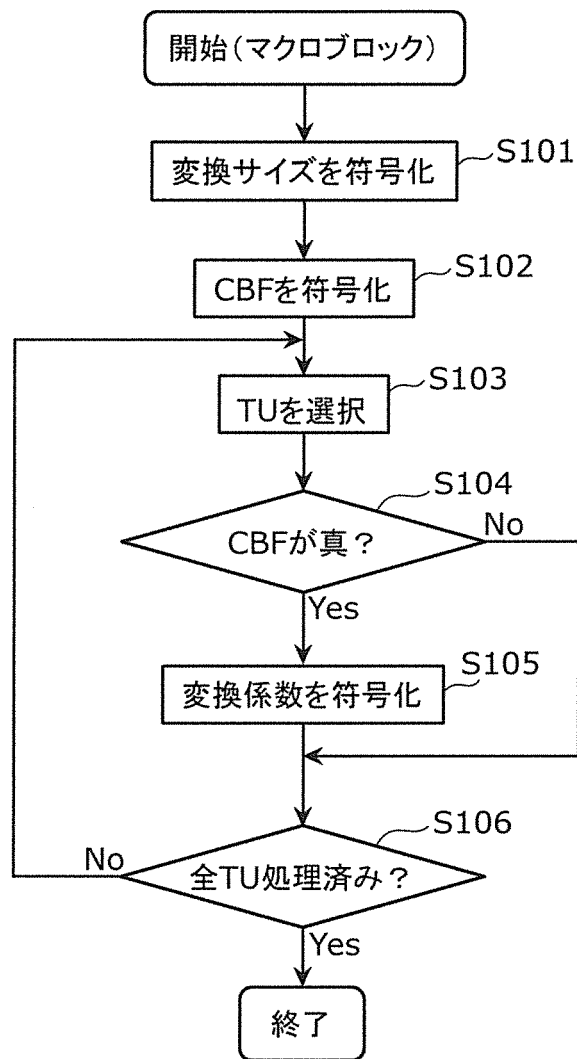
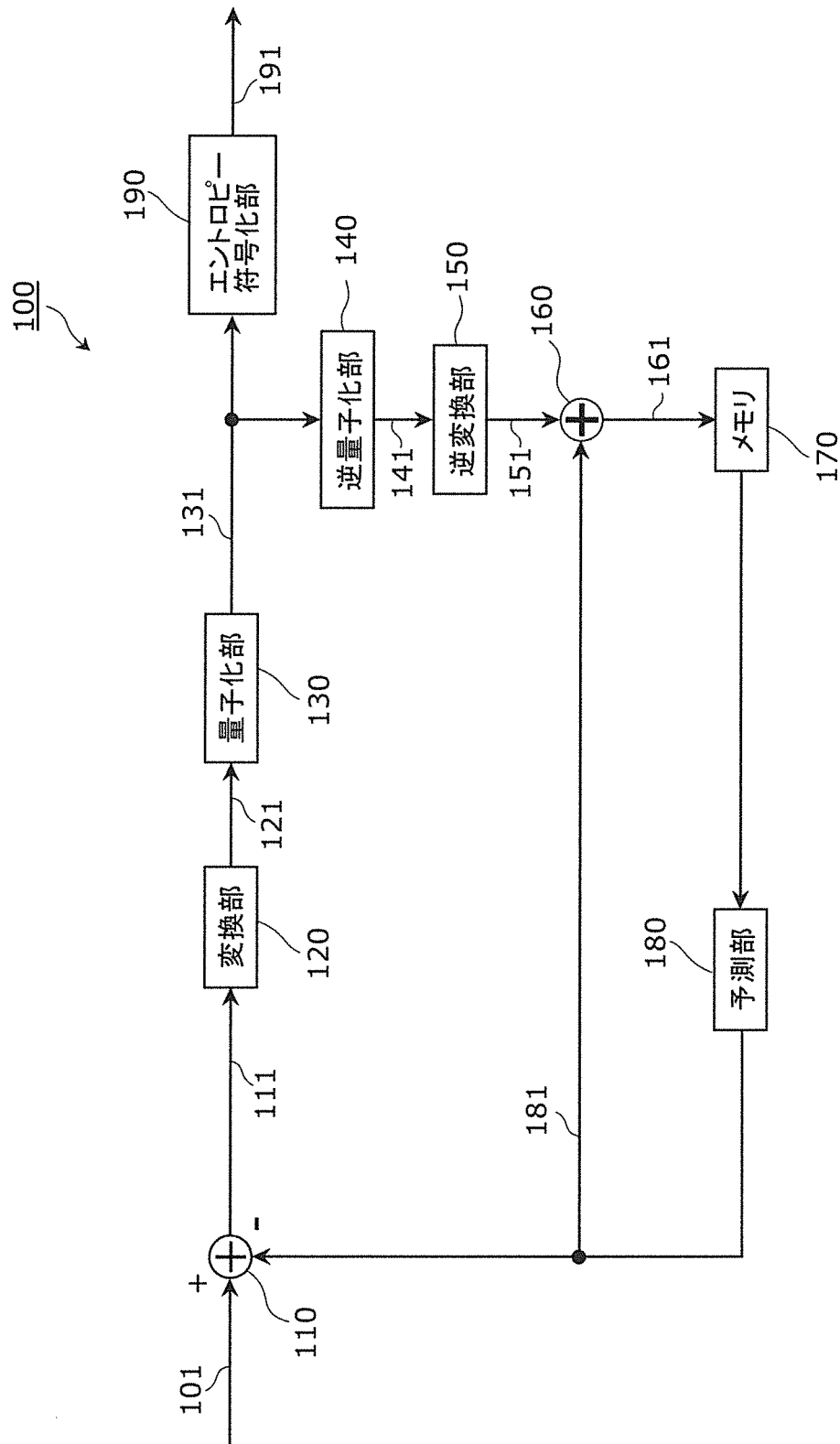


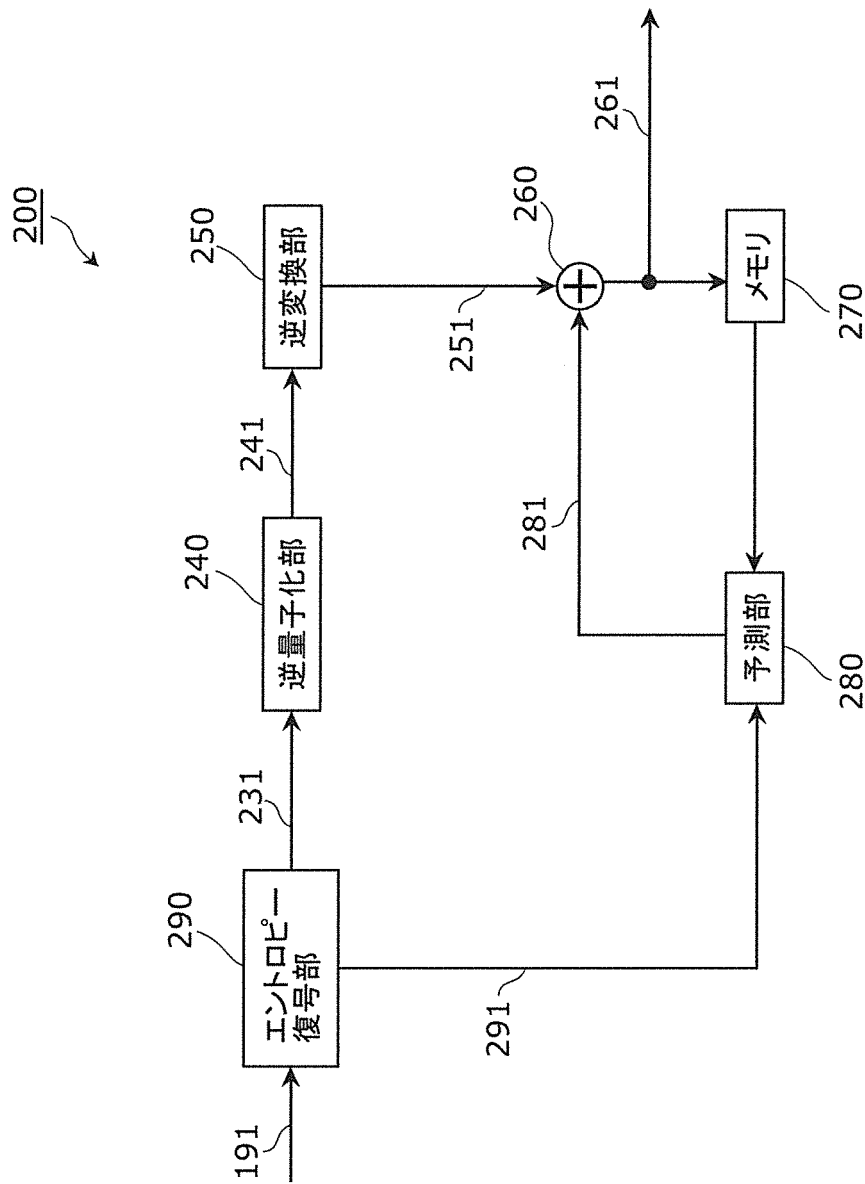
[図1]



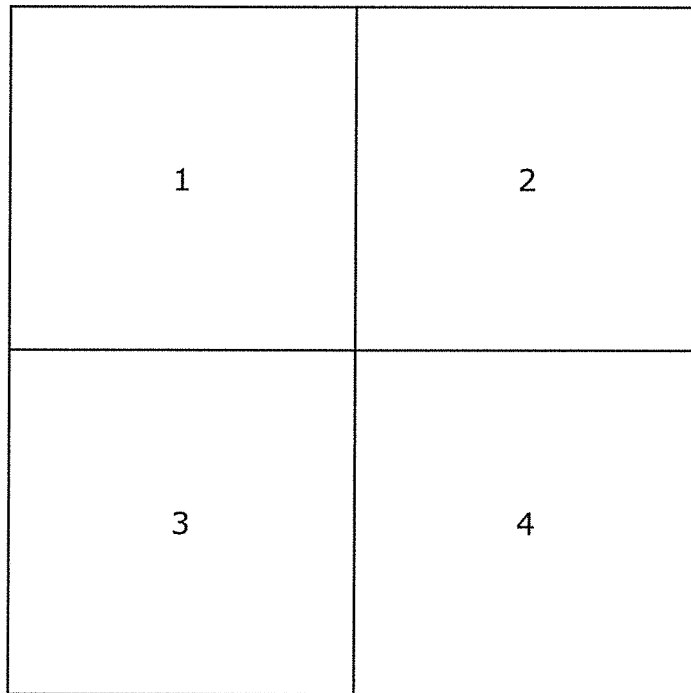
[図2]



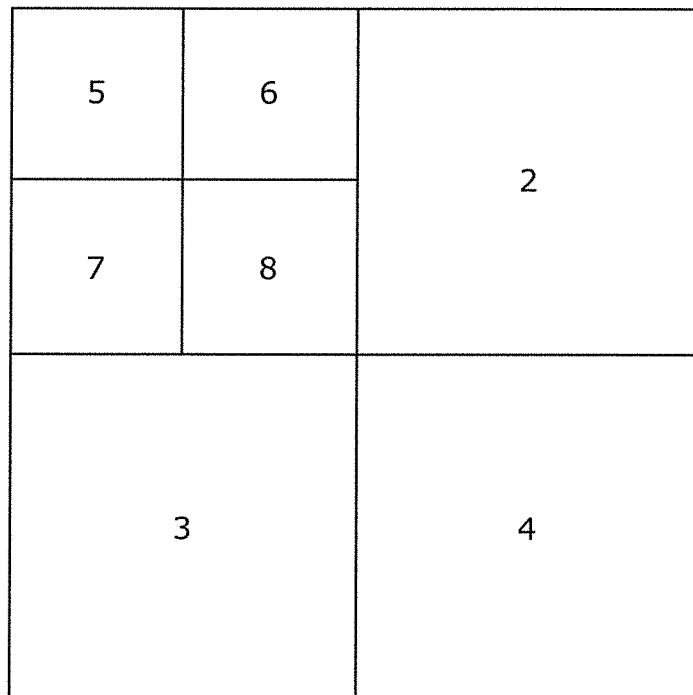
[図3]



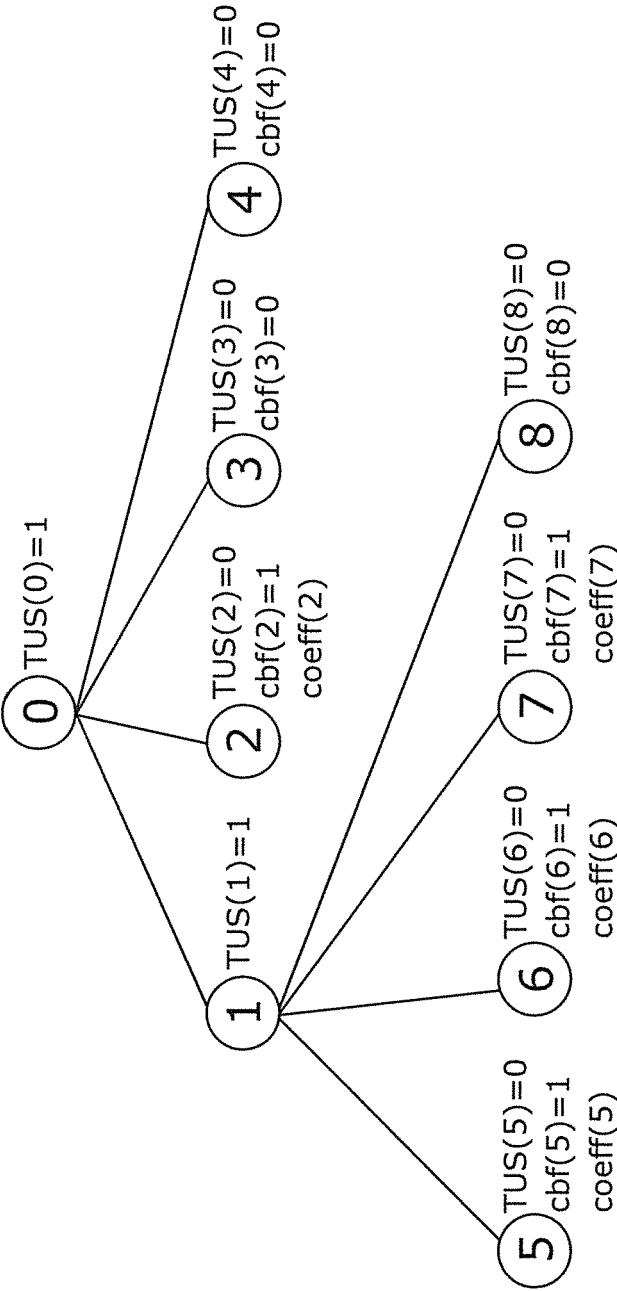
[図4A]



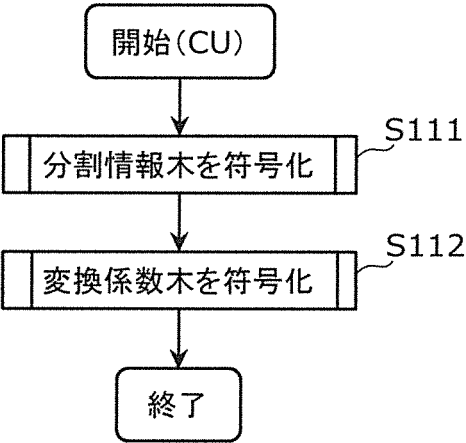
[図4B]



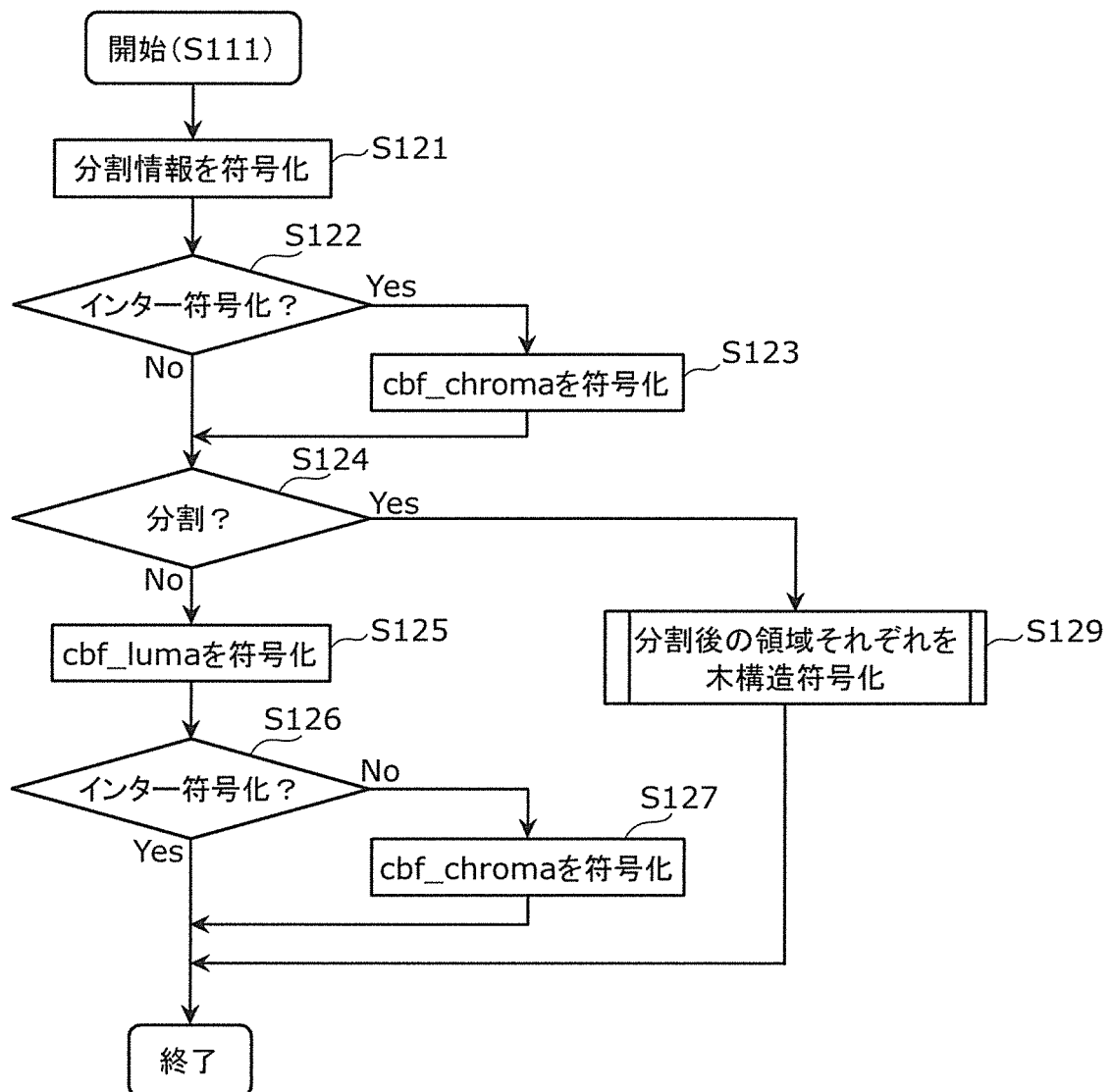
[図5]



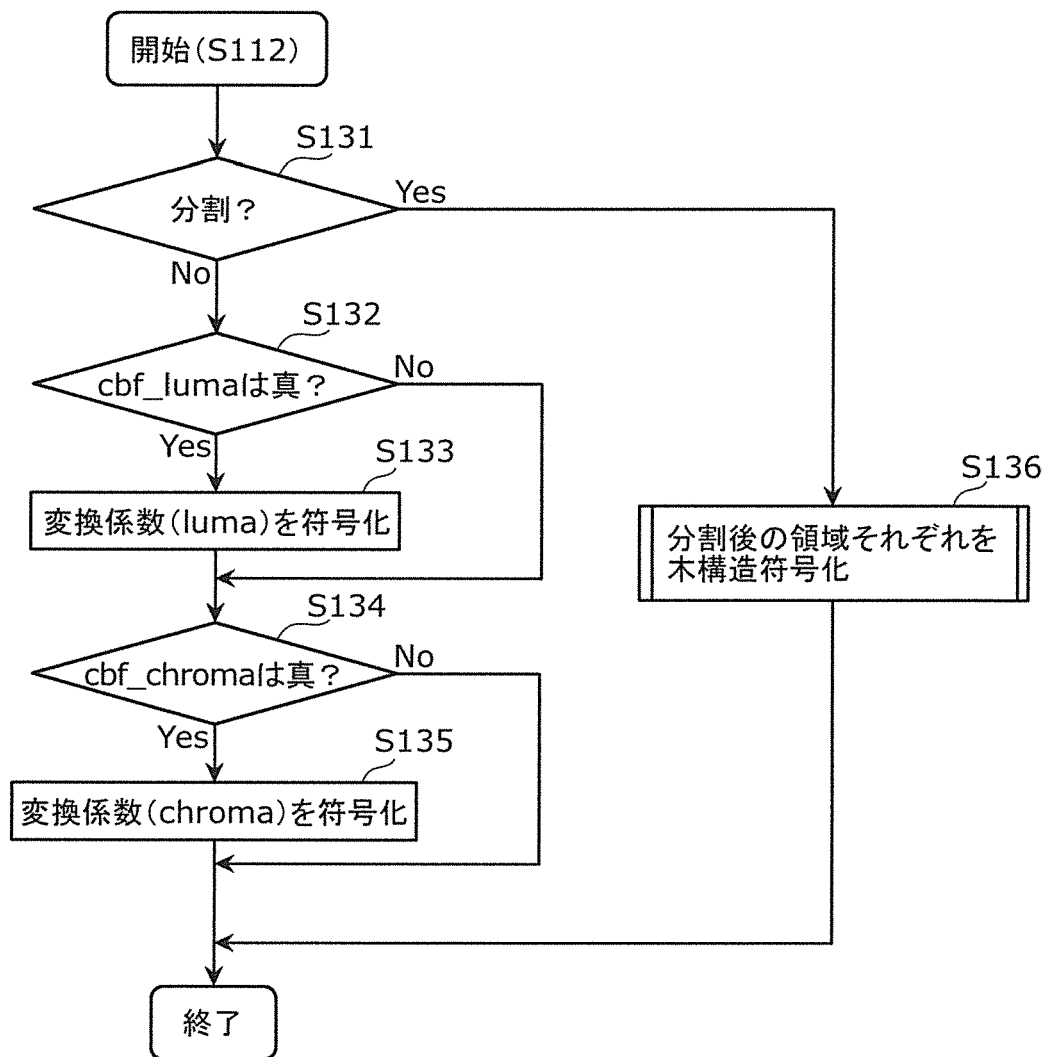
[図6]



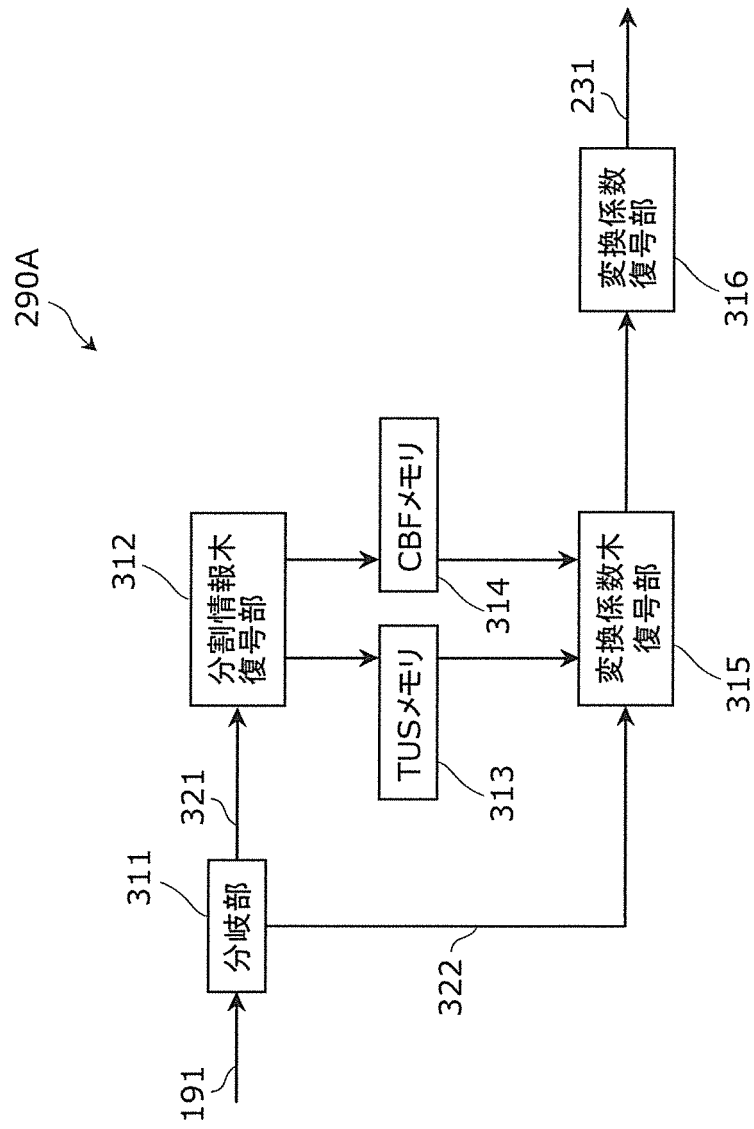
[図7]



[図8]



[図9]



[図10A]

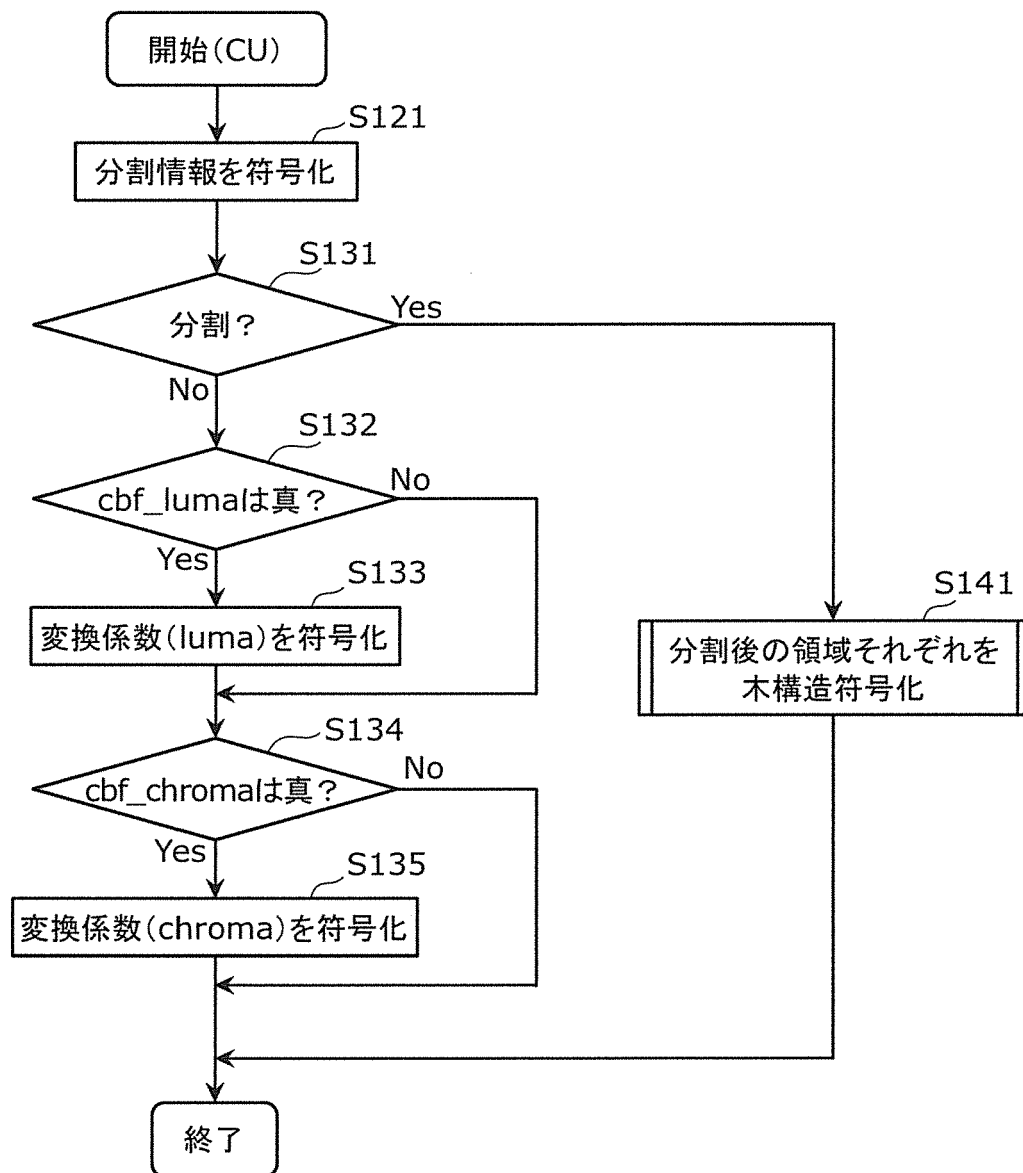
| 分割情報木                                                     | 変換係数木                 |
|-----------------------------------------------------------|-----------------------|
| TUS(0)=1,TUS(1)=1,TUS(5)=0,cbf(5)=1,TUS(6)=0,cbf(6)=1,... | coeff(5),coeff(6),... |

[図10B]

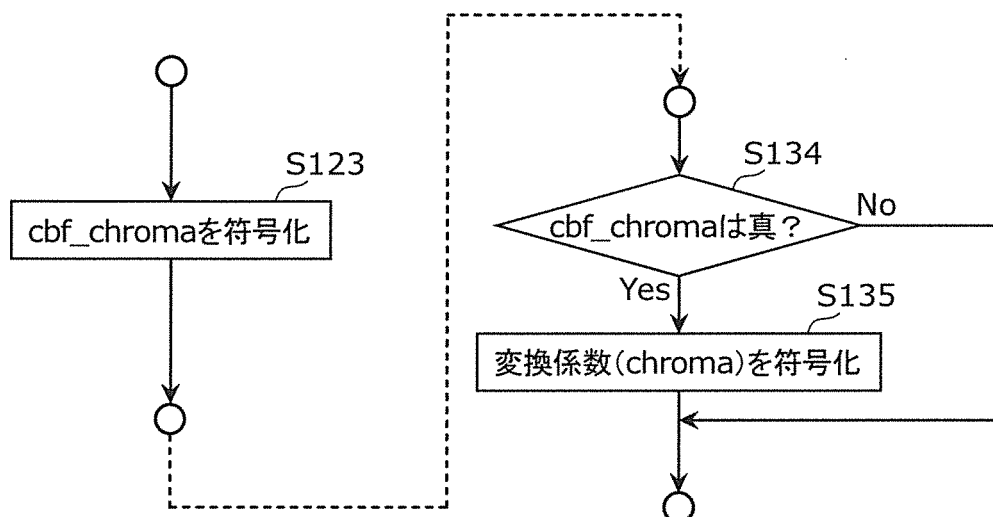
変換統一木

$$\text{TUS}(0)=1, \text{TUS}(1)=1, \text{TUS}(5)=0, \text{cbf}(5)=1, \text{coeff}(5), \text{TUS}(6)=0, \text{cbf}(6)=1, \text{coeff}(6), \dots$$

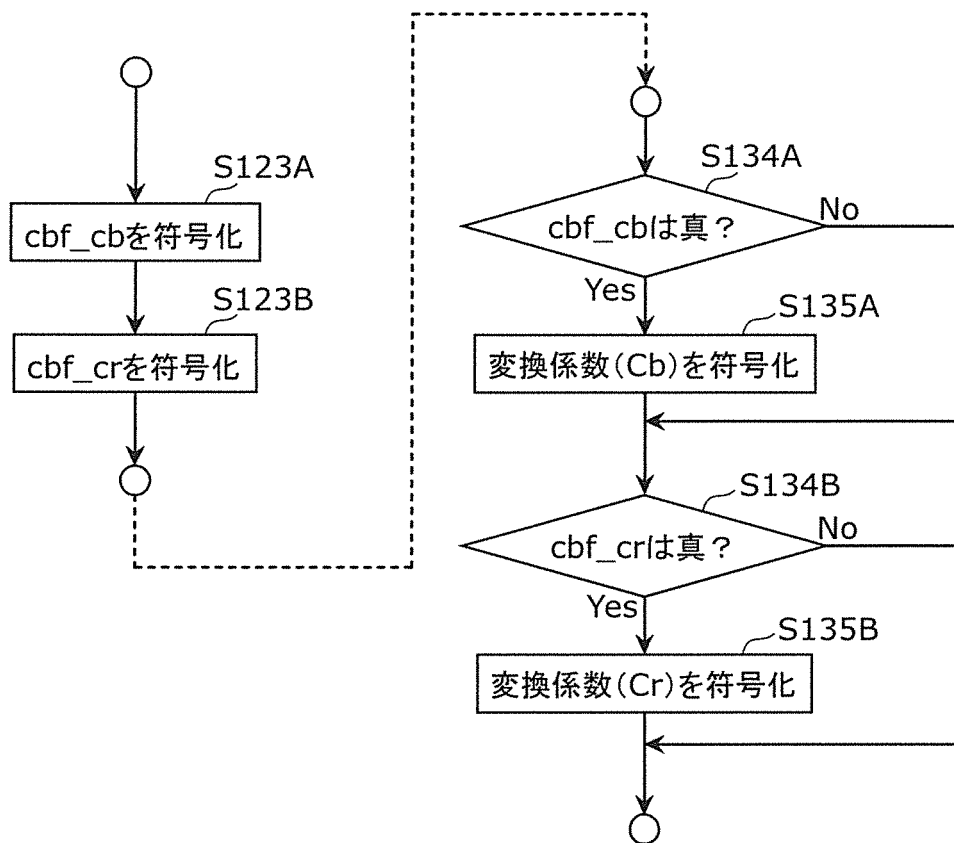
[図11]



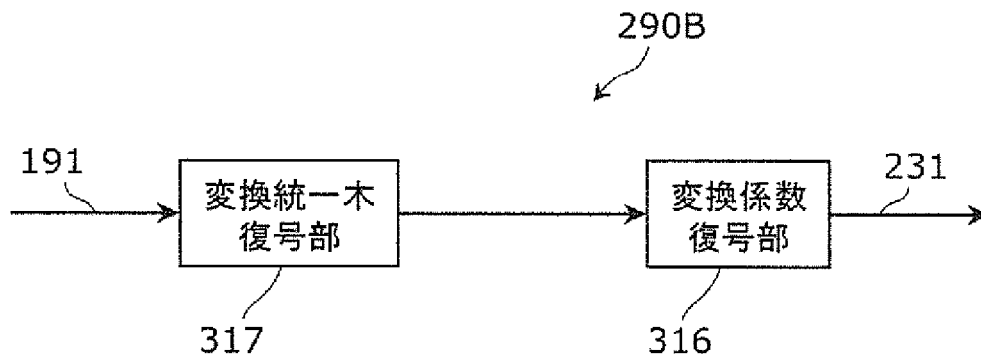
[図12A]



[図12B]



[図13]



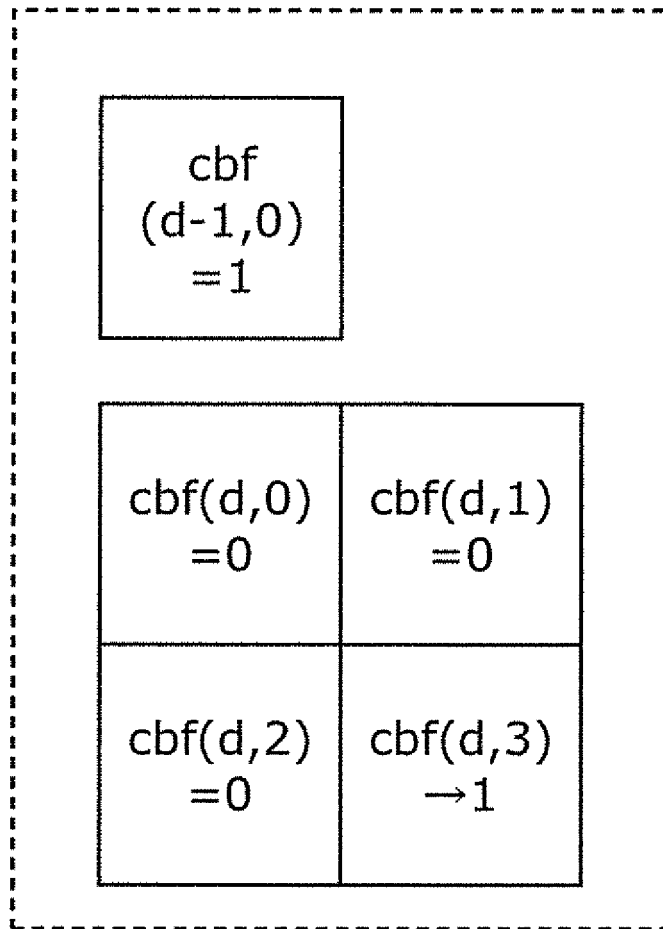
[図14A]

|        |        |
|--------|--------|
| cbf(0) | cbf(1) |
| cbf(2) | cbf(3) |

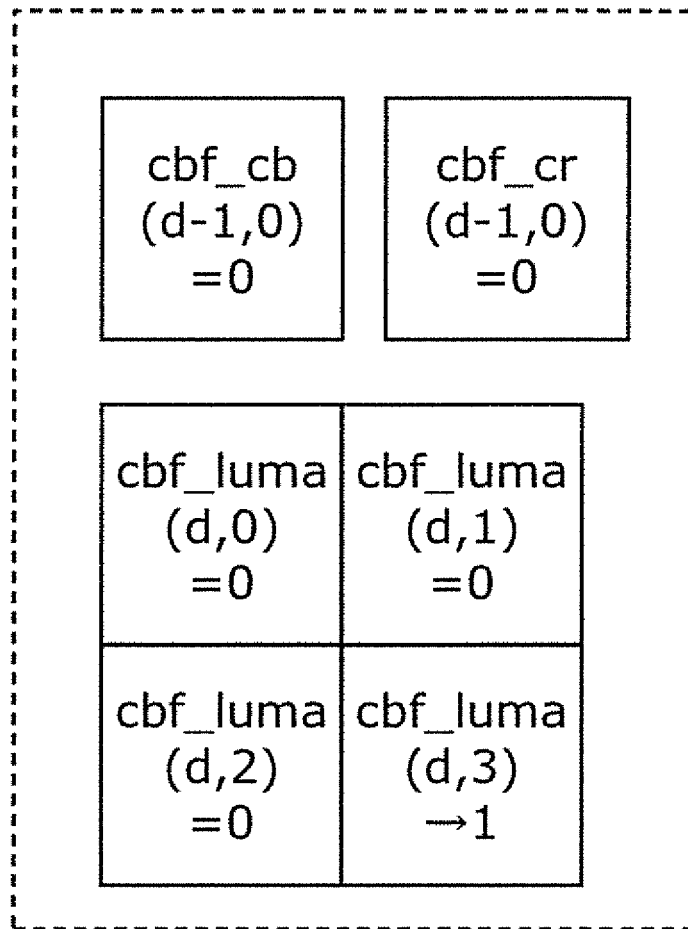
[図14B]

|              |              |
|--------------|--------------|
| cbf(0)<br>=0 | cbf(1)<br>=0 |
| cbf(2)<br>=0 | cbf(3)<br>→1 |

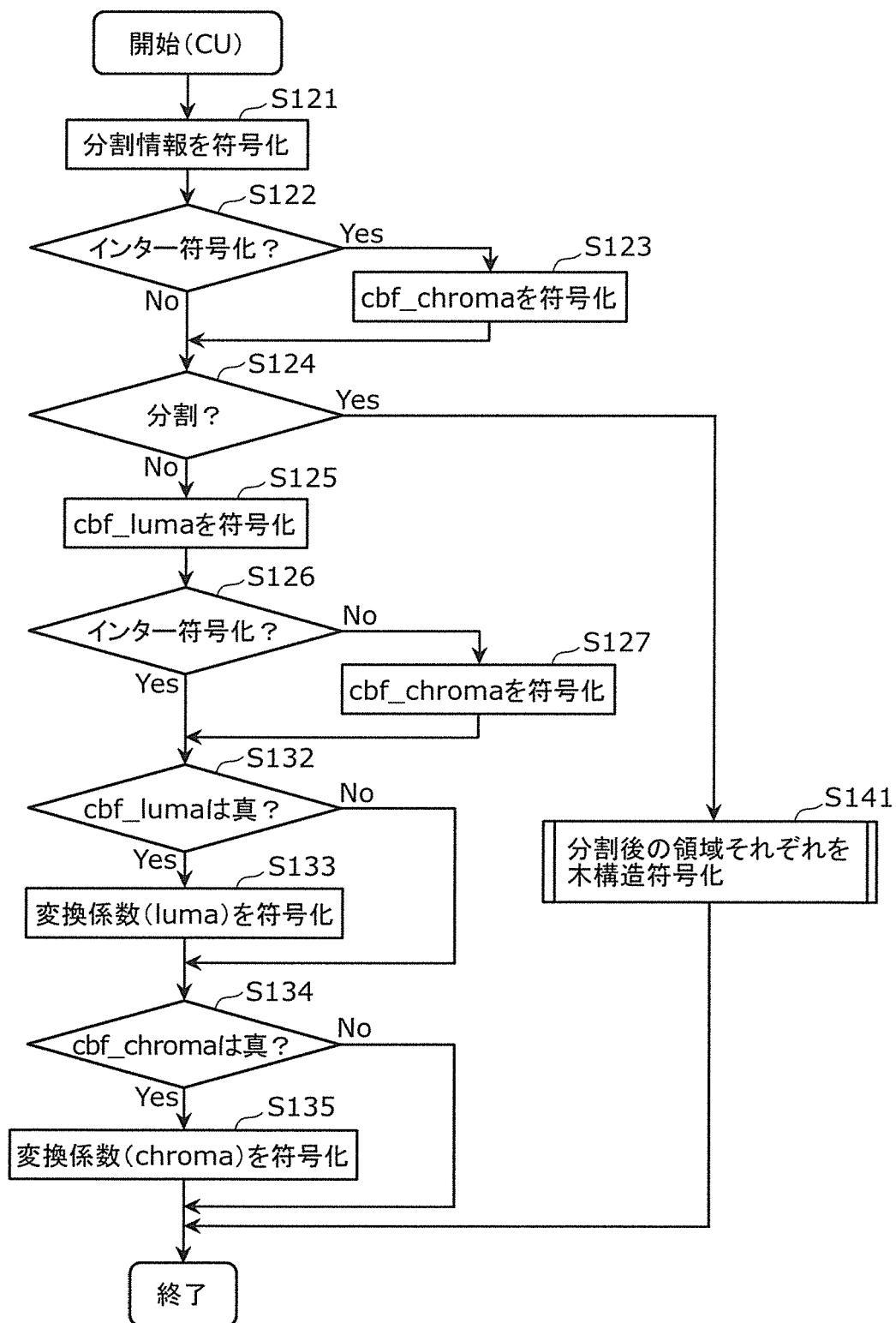
[図14C]



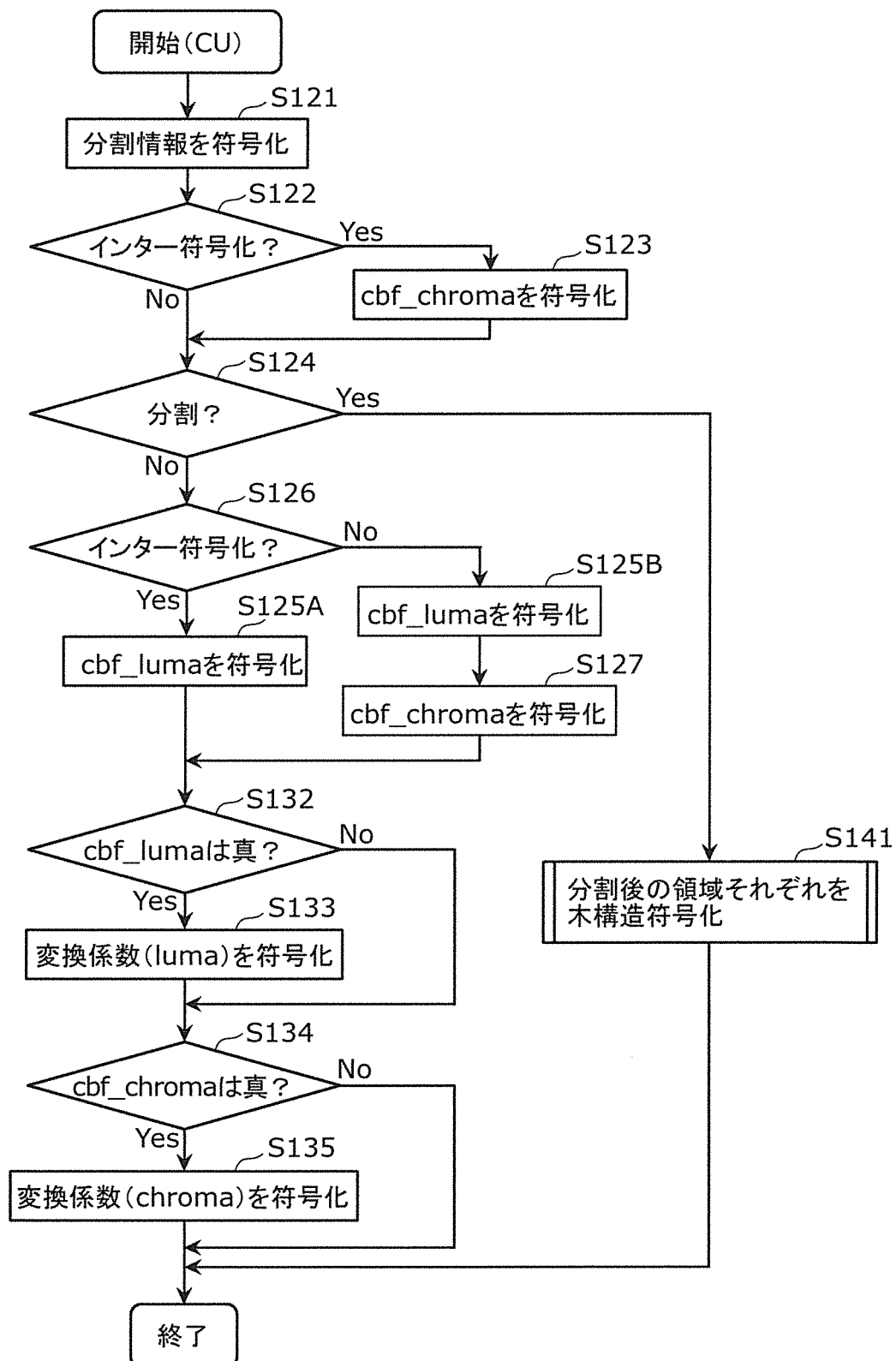
[図14D]



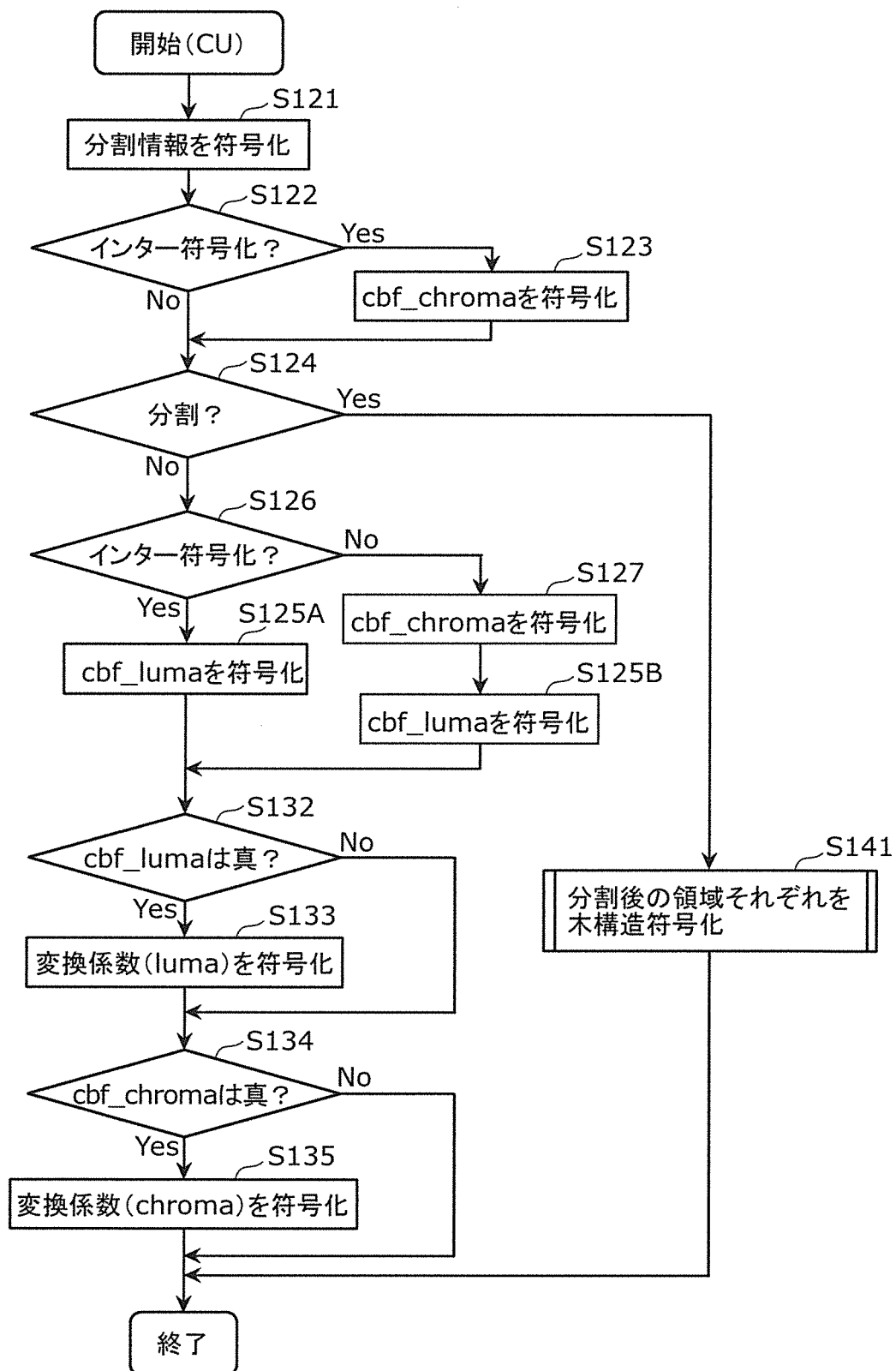
[図15]



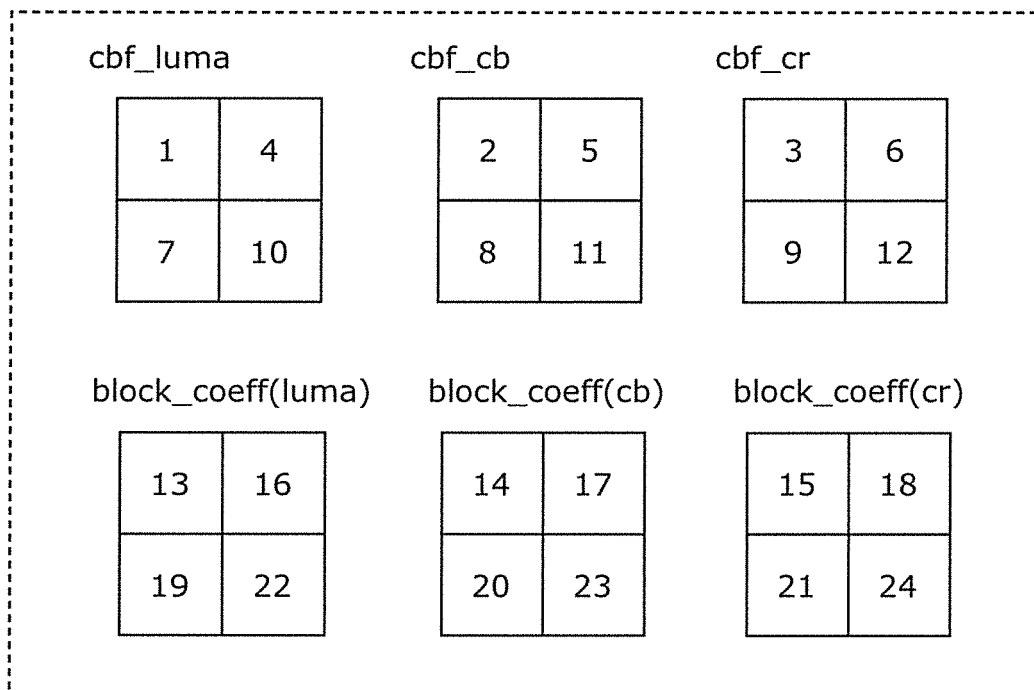
[図16]



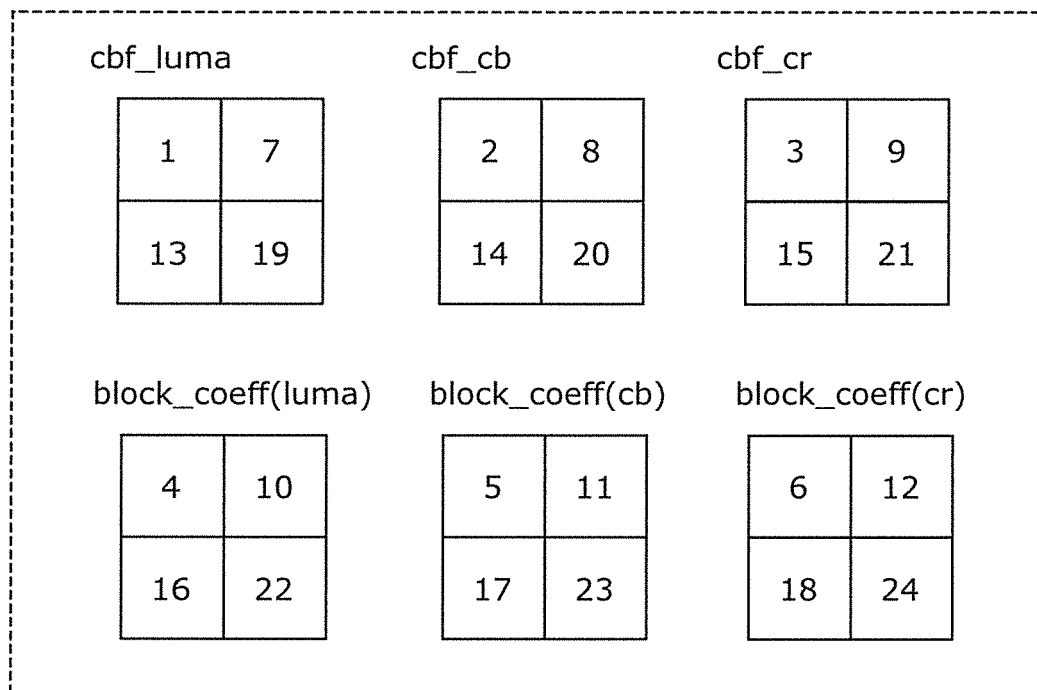
[図17]



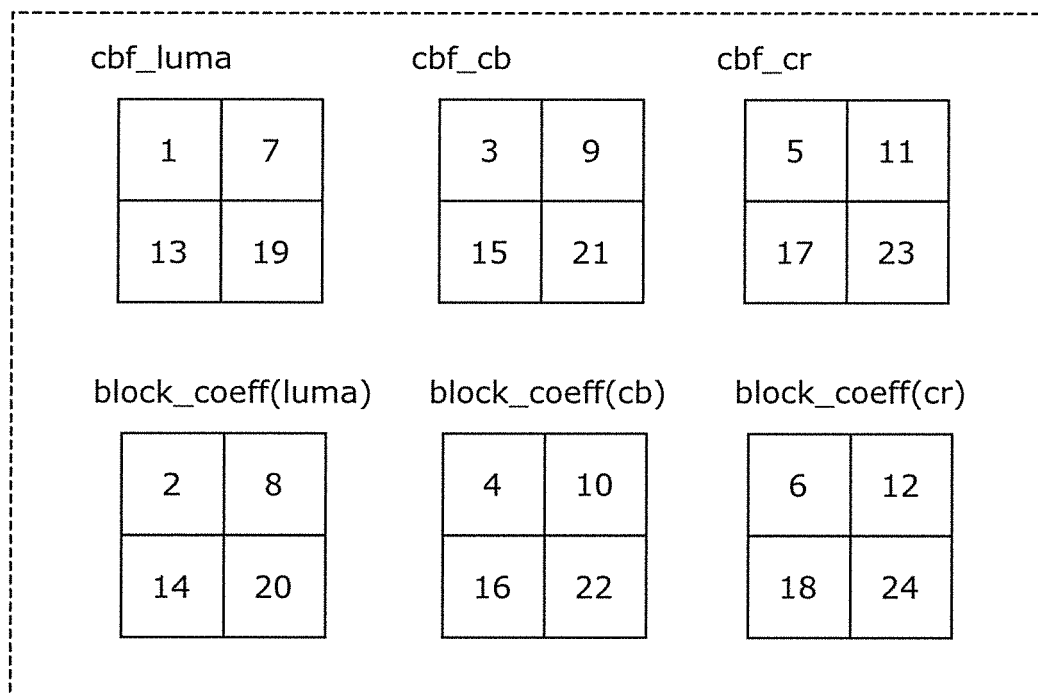
[図18A]



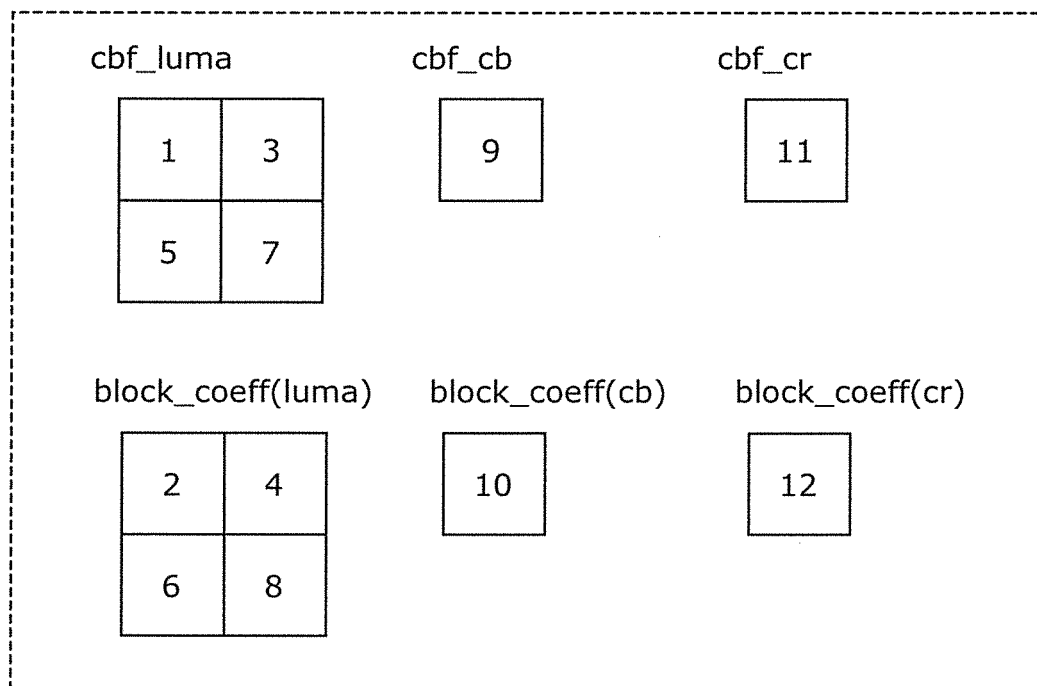
[図18B]



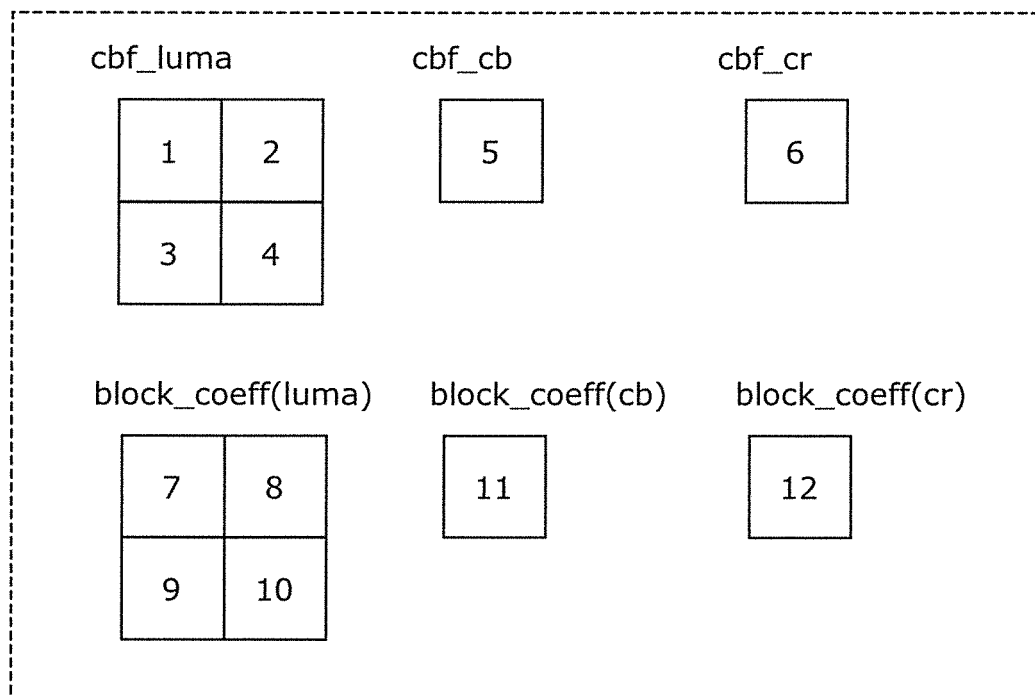
[図18C]



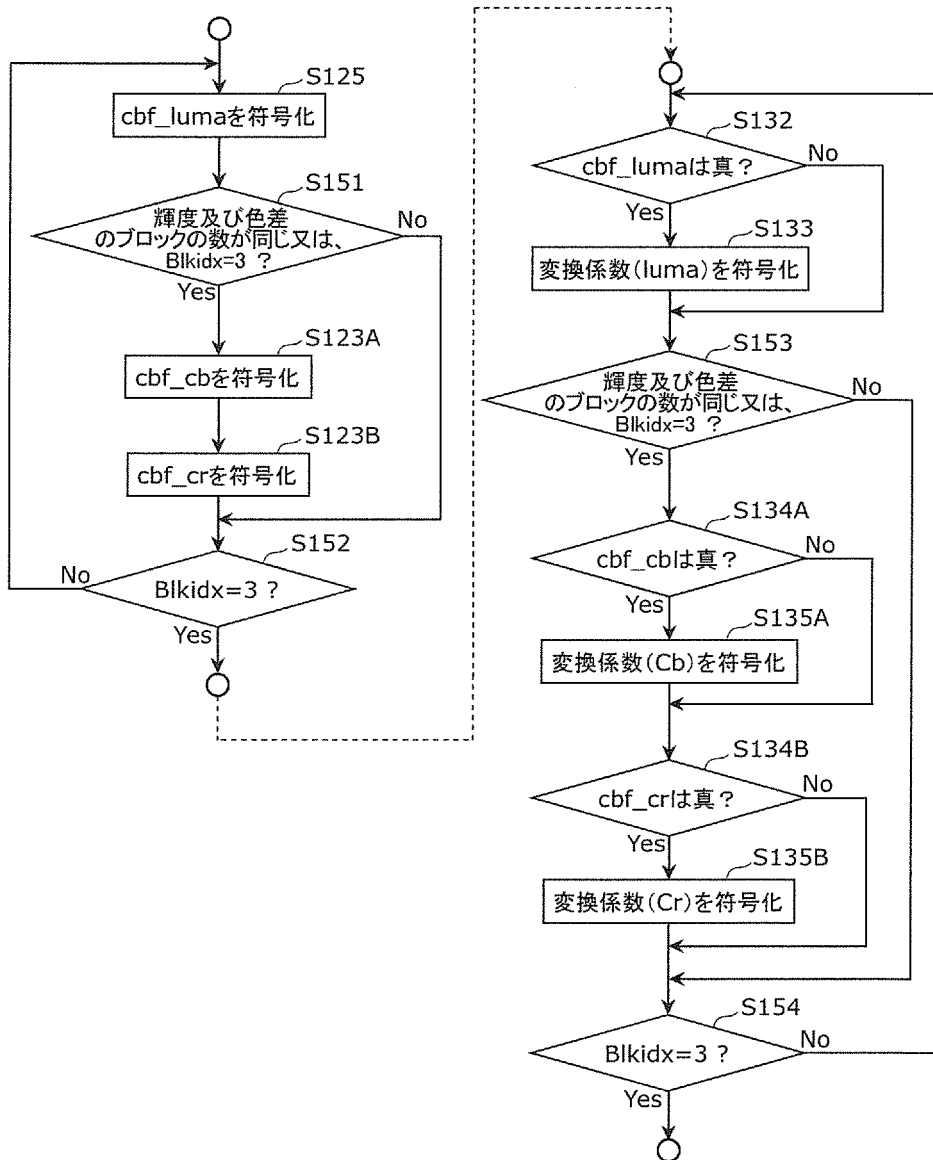
[図19A]



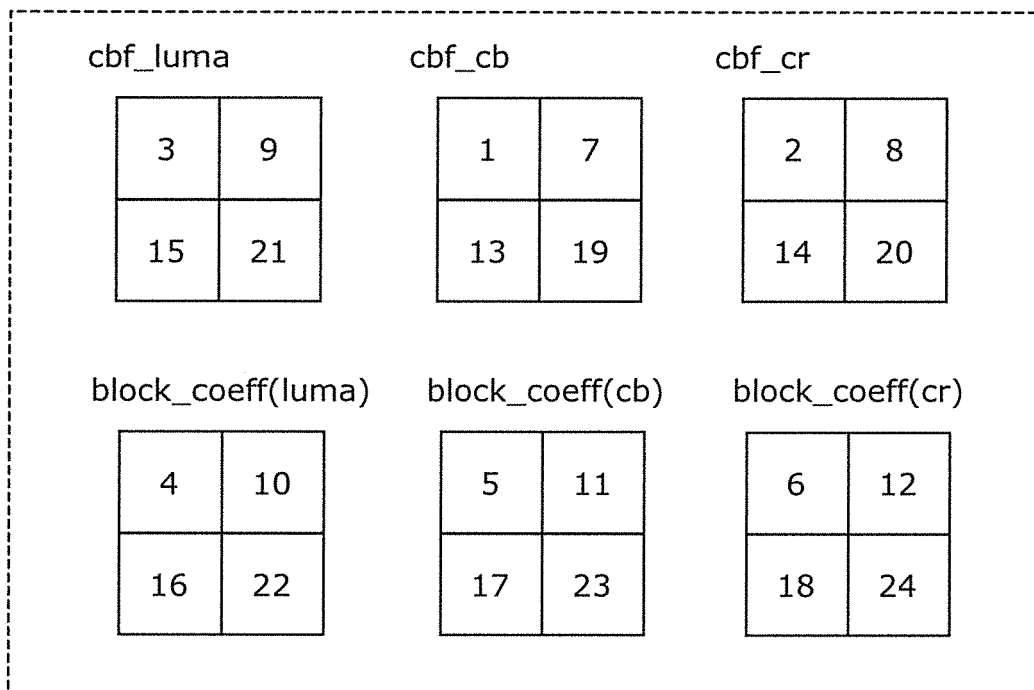
[図19B]



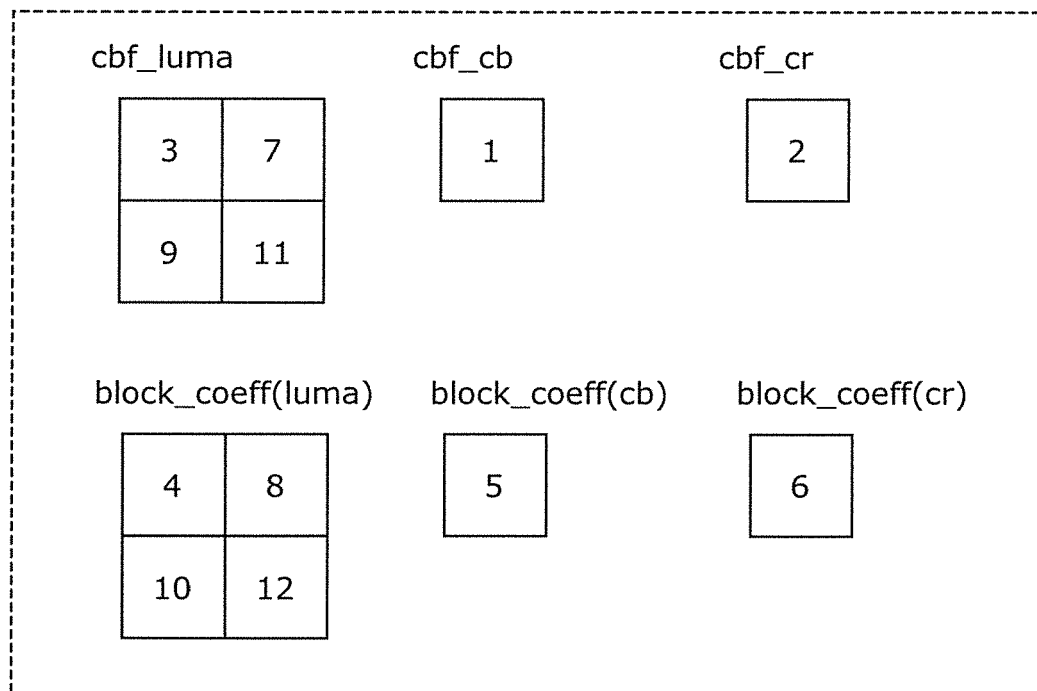
[図20]



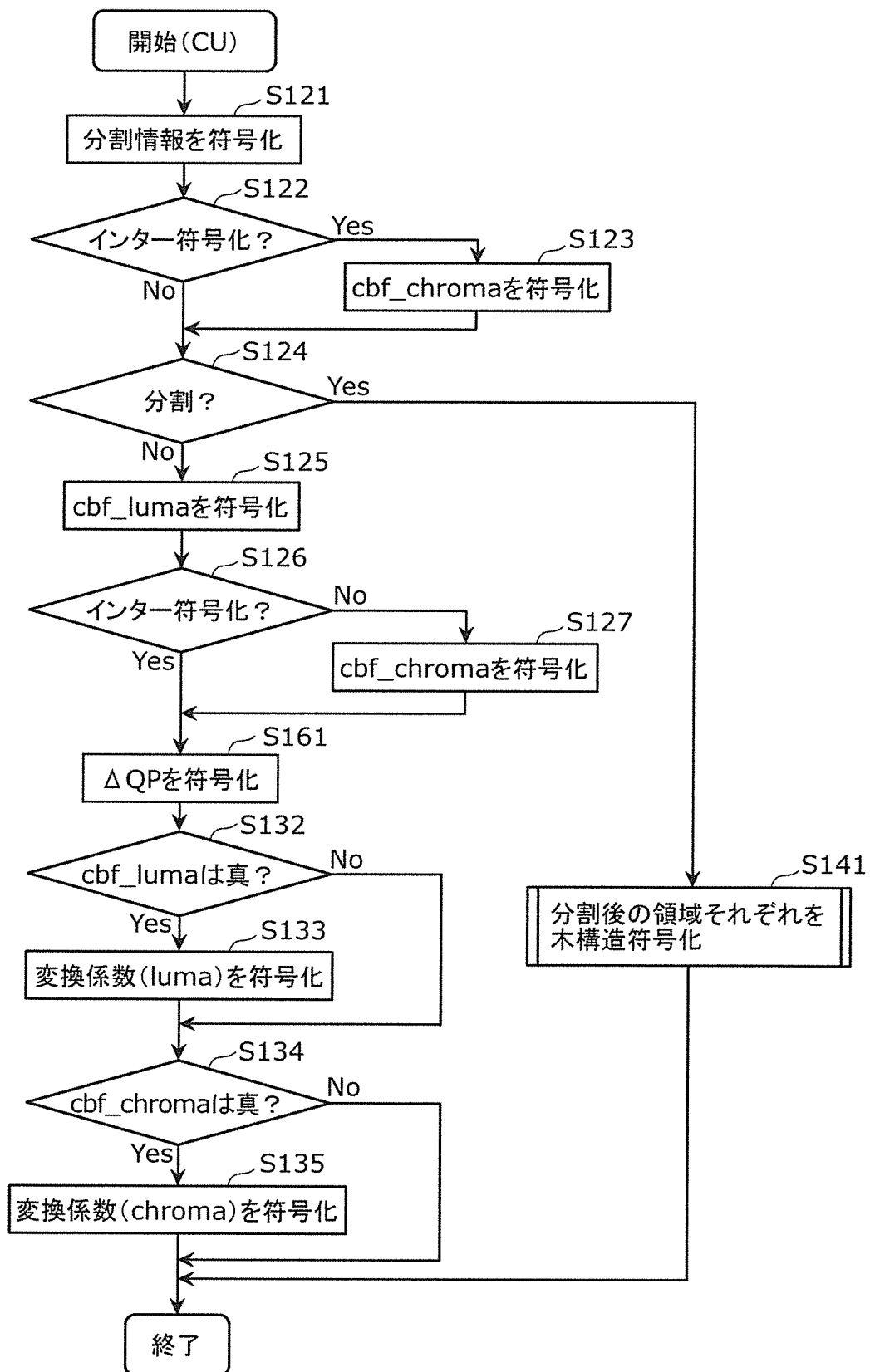
[図21A]



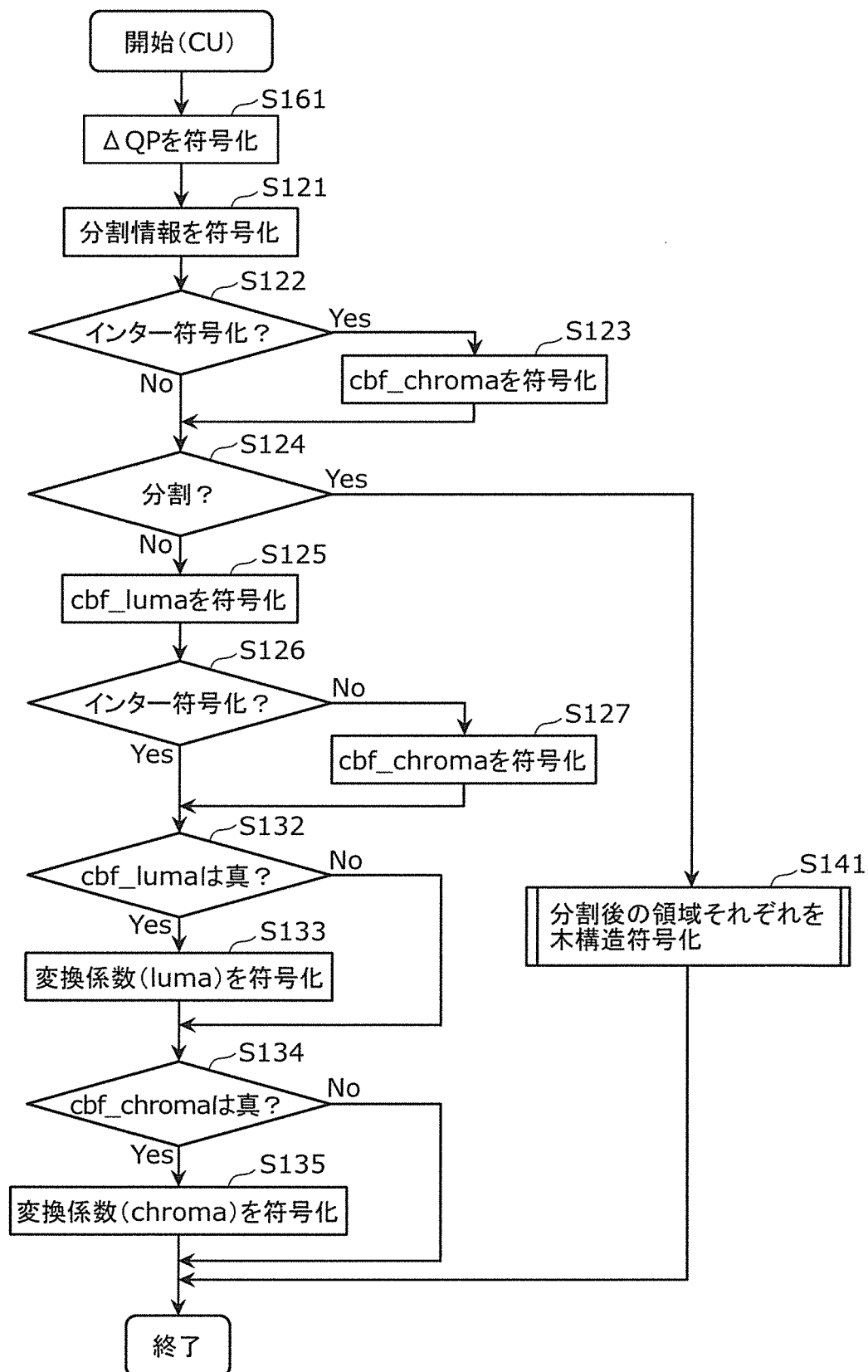
[図21B]



[図22A]



[図22B]



[図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 ] )
    {
        4分割して再帰呼び出し
    } 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 )
    }
}

```

## [図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 ]
                    }
                }
            }
        }
    }
}

```

[図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 ]
    }
}
      .
      .
      .

```

• • •

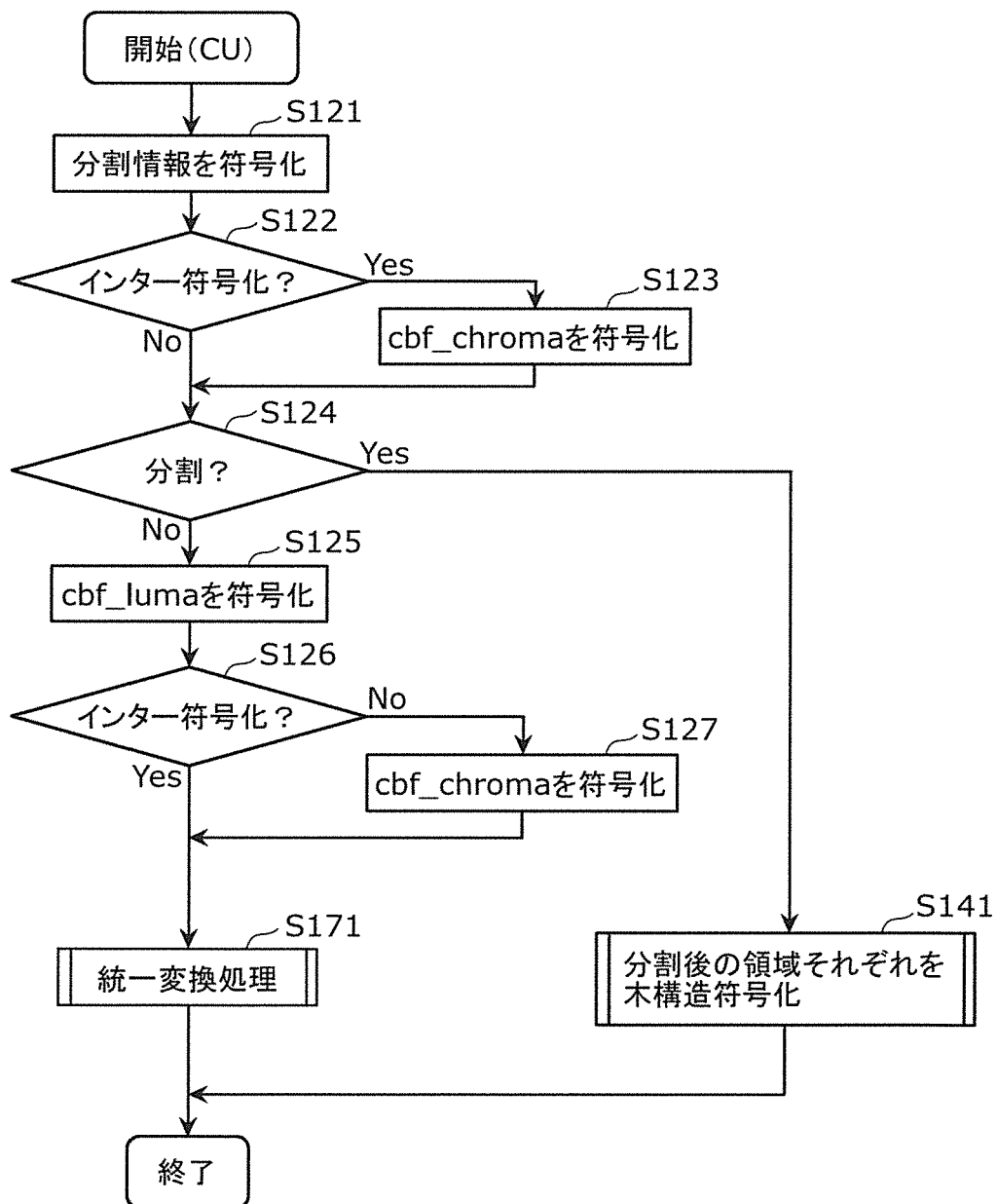
```

{
    if( cbf_luma[ x0 ][ y0 ][ trafoDepth ] || cbf_cb[ x0 ][ y0 ][ trafoDepth ] || cbf_cr[ x0 ][ y0 ][ trafoDepth ] )
    {
        if( cu_qp_delta_enabled_flag && !isCuQpDeltaCoded ) {
            cu_qp_delta
                IsCuQpDeltaCoded = 1
        }
    }
}

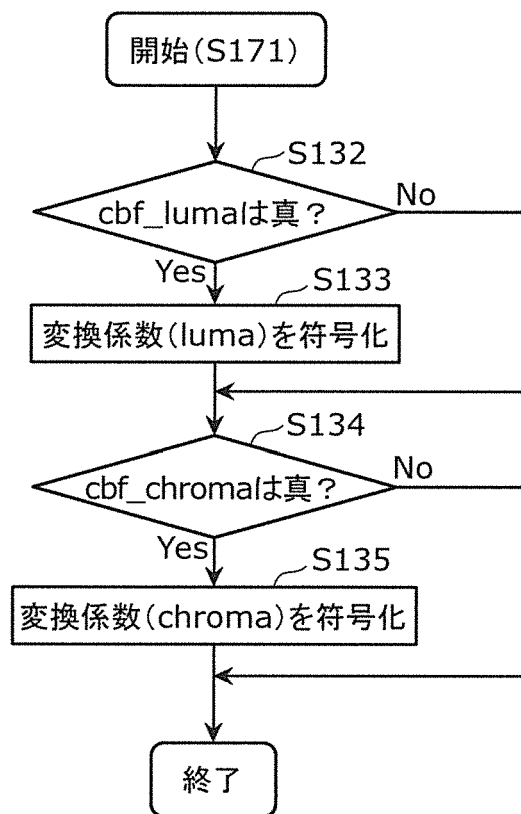
log2TrafoSize = ( log2TrafoWidth + log2TrafoHeight ) >> 1
log2TrafoSizeC = ( log2TrafoSize == Log2MinTrafoSize ) ? log2TrafoSize : ( log2TrafoSize - 1 )
if( cbf_luma[ x0 ][ y0 ][ trafoDepth ] )
    residual_coding_cabac( x0, y0, log2TrafoSize, trafoDepth, scanIdx, cidx 0 )
if(log2TrafoSize > Log2MinTrafoSize)
{
    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 )
} else if( log2TrafoSize == Log2MinTrafoSize && blk == 3 )
{
    if( cbf_cb[ xC ][ yC ][ trafoDepth ] )
        residual_coding_cabac( xC, yC, log2TrafoSize, trafoDepth, scanIdxC, 1 )
    if( cbf_cr[ xC ][ yC ][ trafoDepth ] )
        residual_coding_cabac( xC, yC, log2TrafoSize, trafoDepth, scanIdxC, 2 )
}
}
}
}
}

```

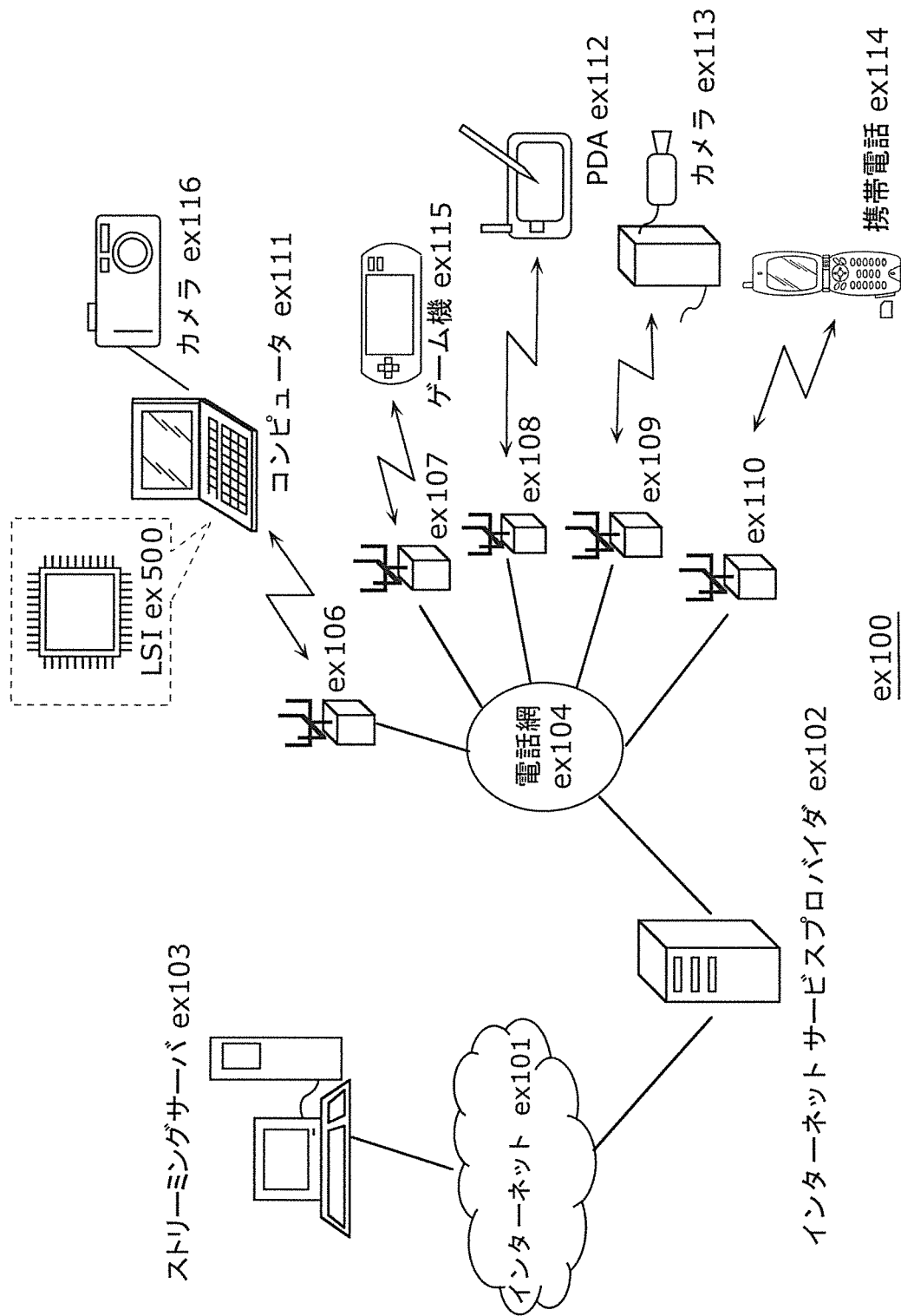
[図25A]



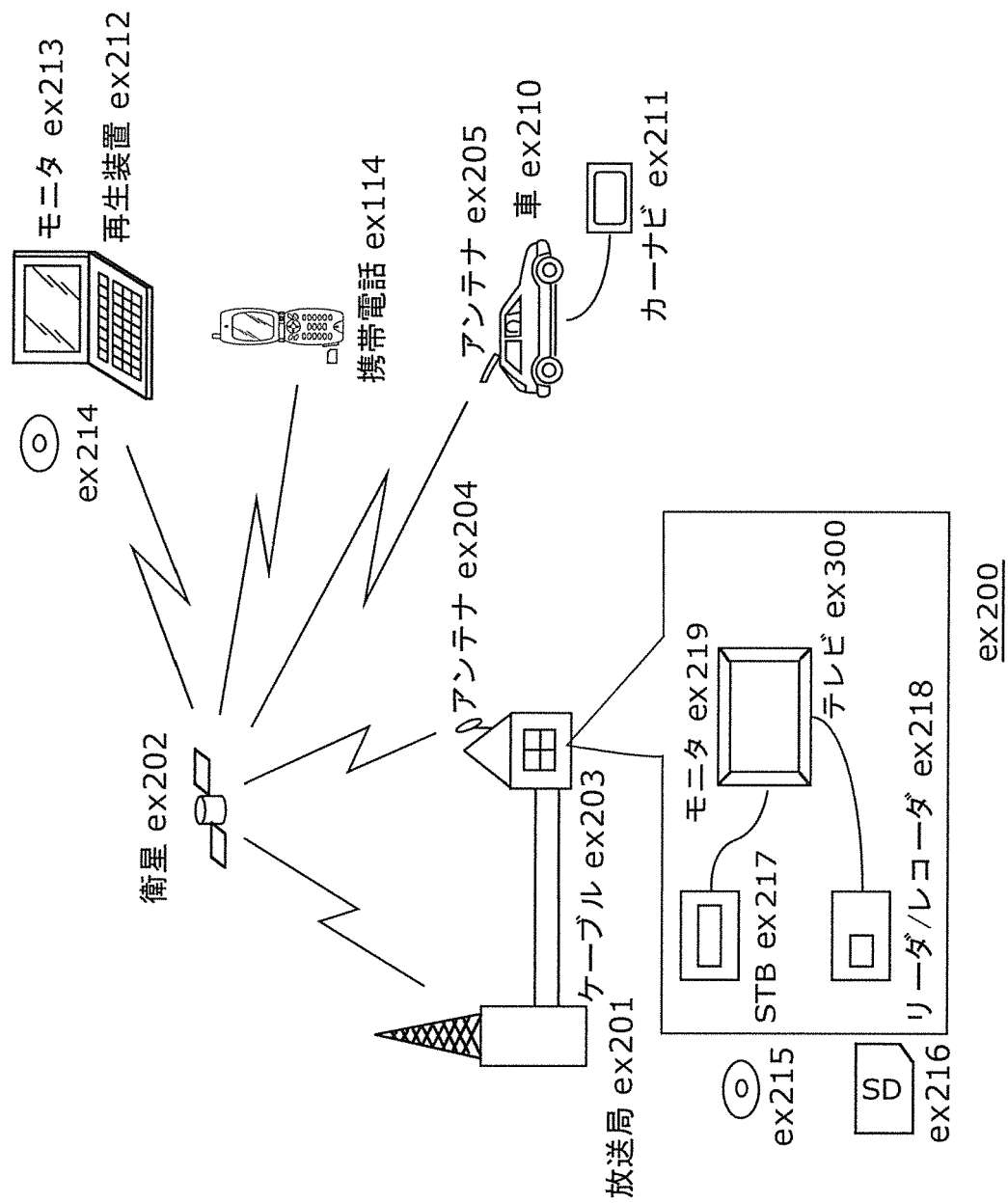
[図25B]



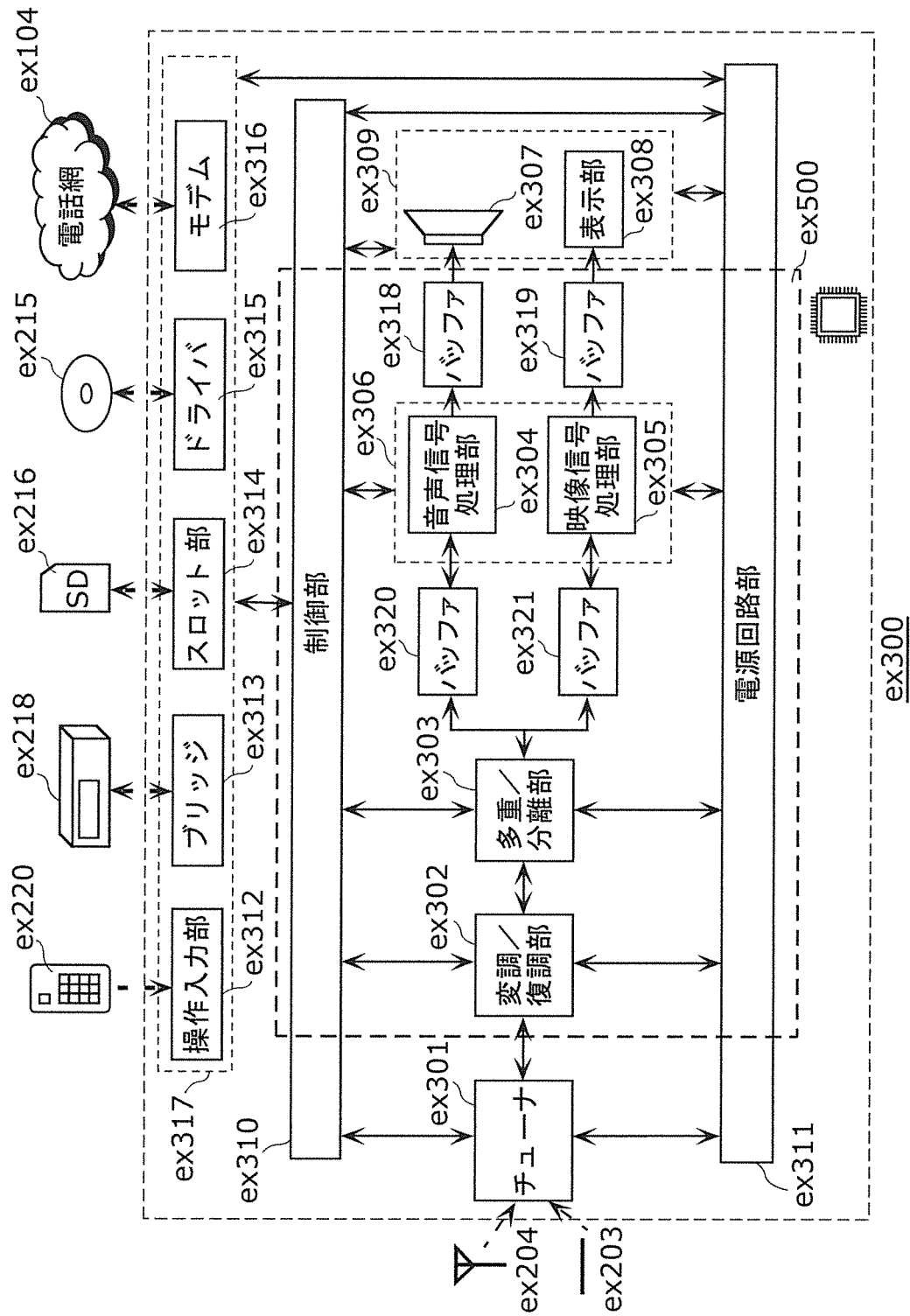
[図26]



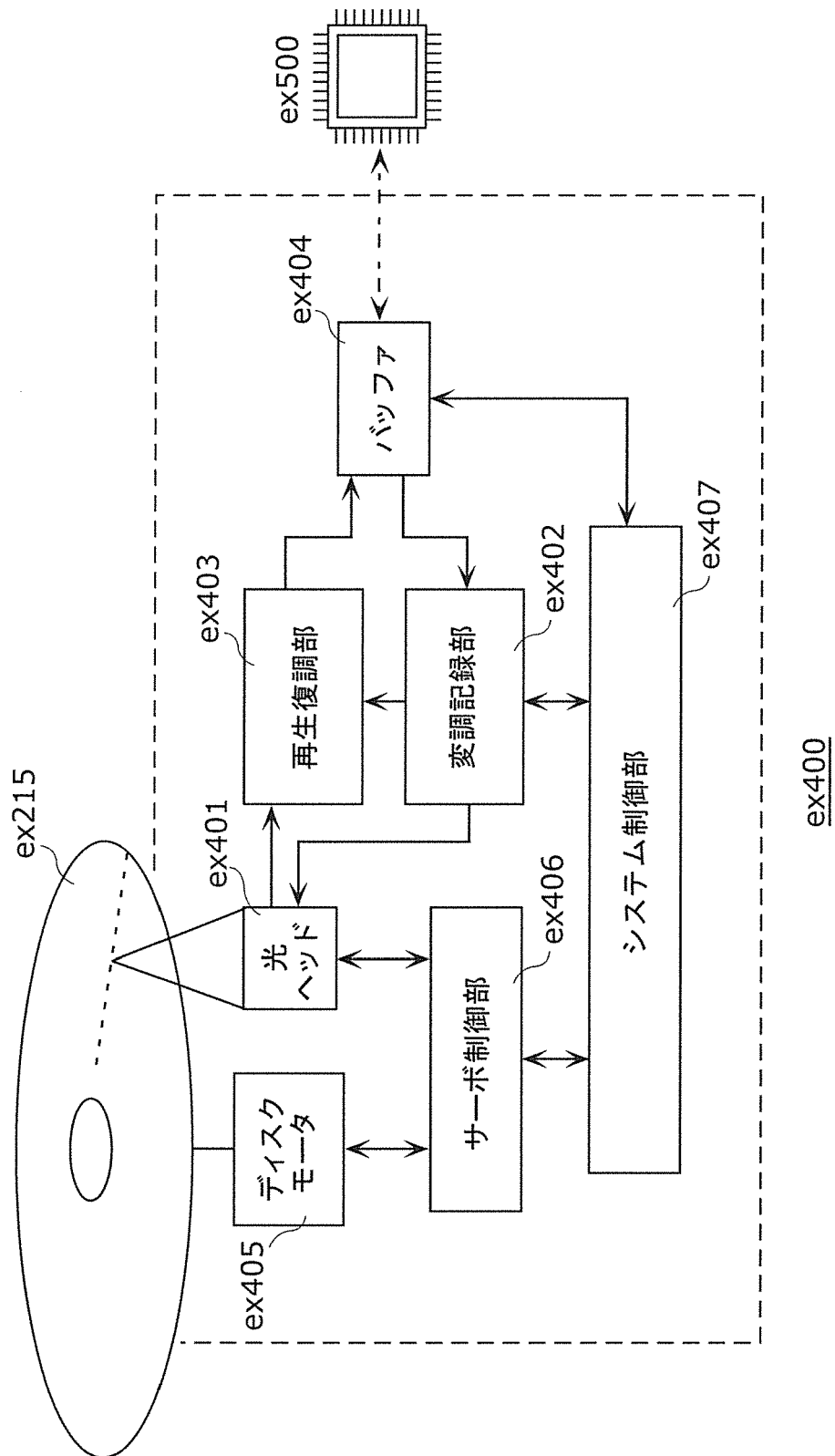
[図27]



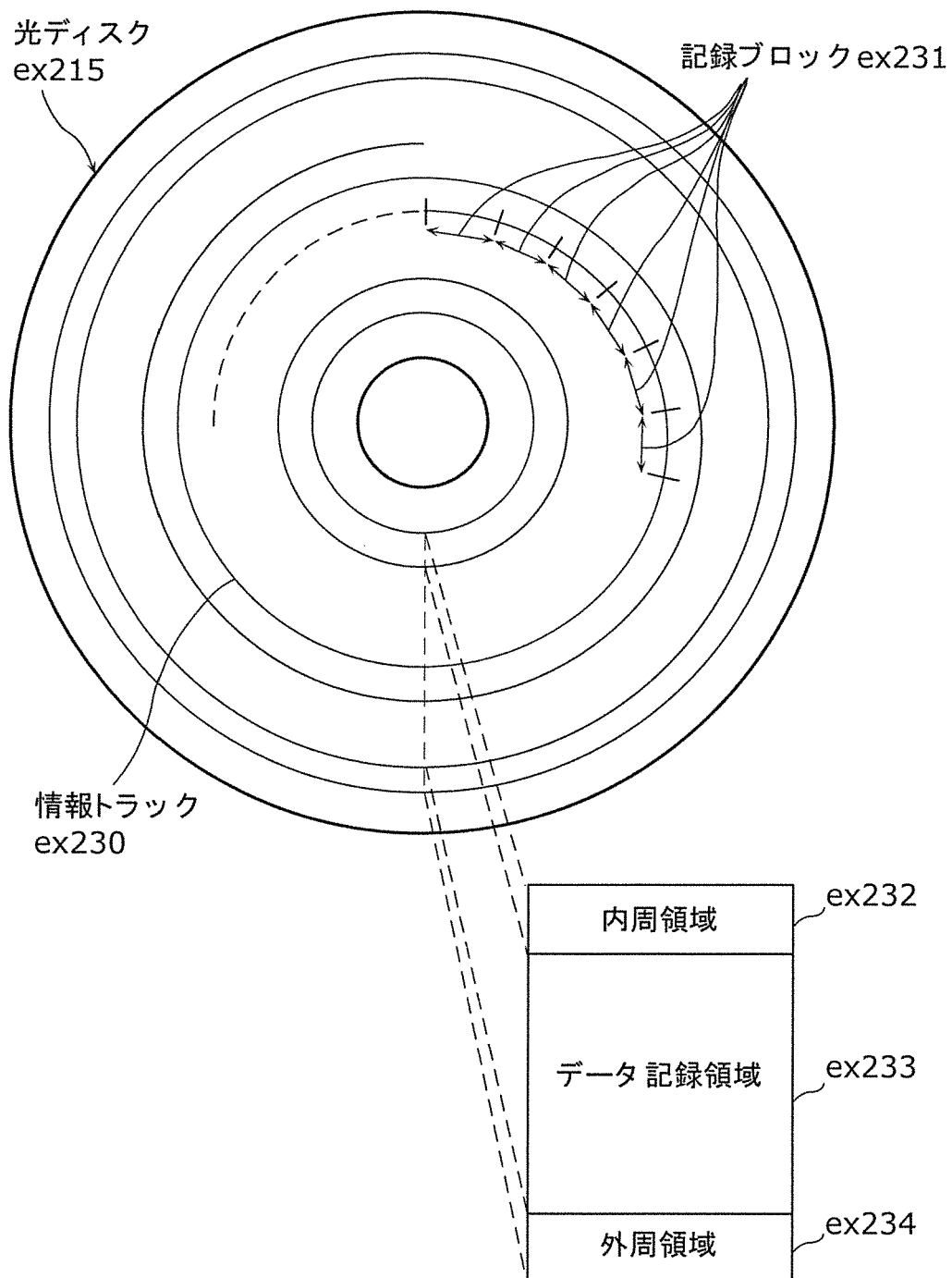
[図28]



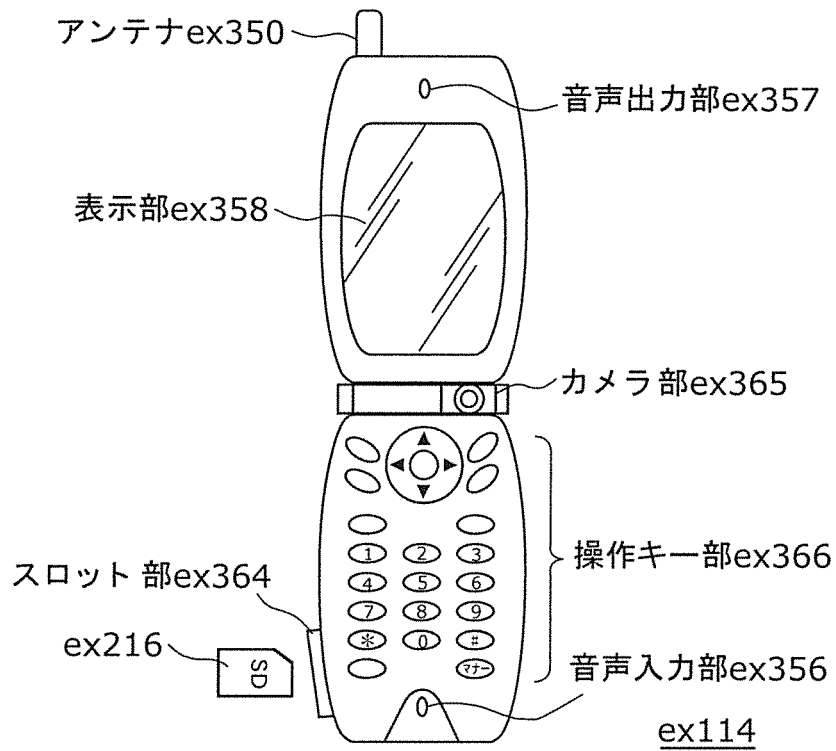
[図29]



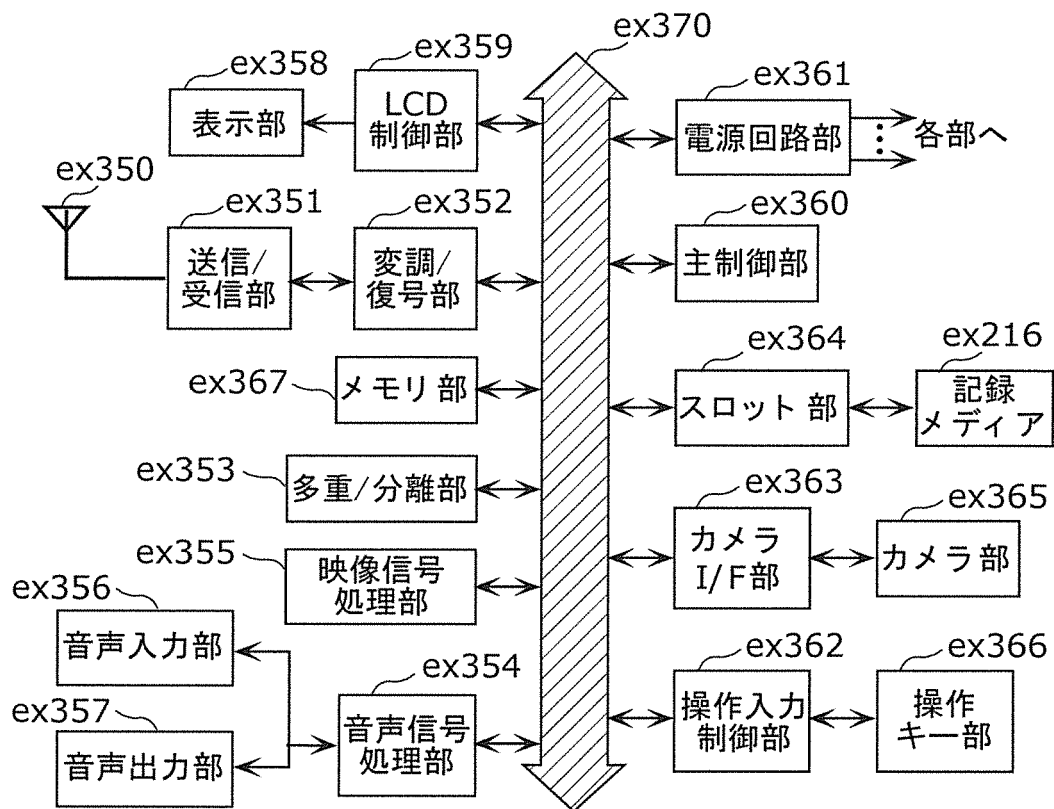
[図30]



[図31A]



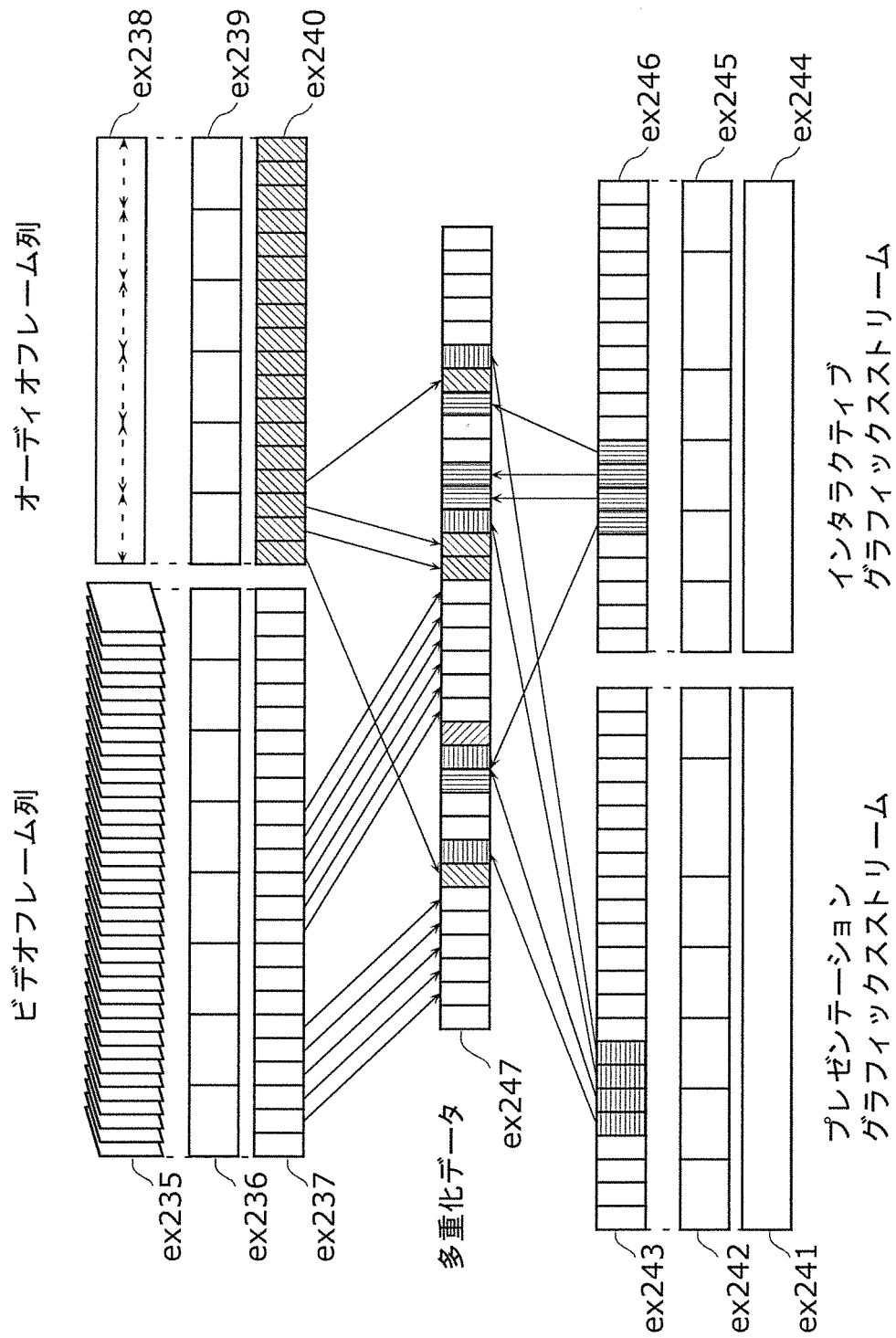
[図31B]



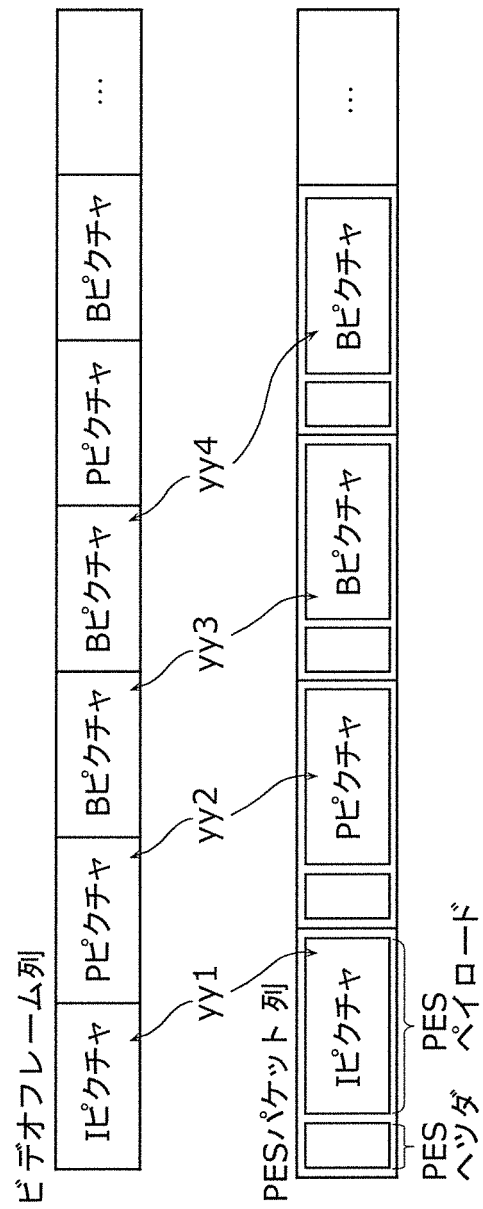
[図32]

|                                   |
|-----------------------------------|
|                                   |
| ビデオストリーム(PID=0x1011 主映像)          |
|                                   |
| オーディオストリーム(PID=0x1100)            |
|                                   |
| オーディオストリーム(PID=0x1101)            |
|                                   |
| プレゼンテーショングラフィックスストリーム(PID=0x1200) |
|                                   |
| プレゼンテーショングラフィックスストリーム(PID=0x1201) |
|                                   |
| インタラクティブグラフィックスストリーム(PID=0x1400)  |
|                                   |
| ビデオストリーム(PID=0x1B00 副映像)          |
|                                   |
| ビデオストリーム(PID=0x1B01 副映像)          |
|                                   |

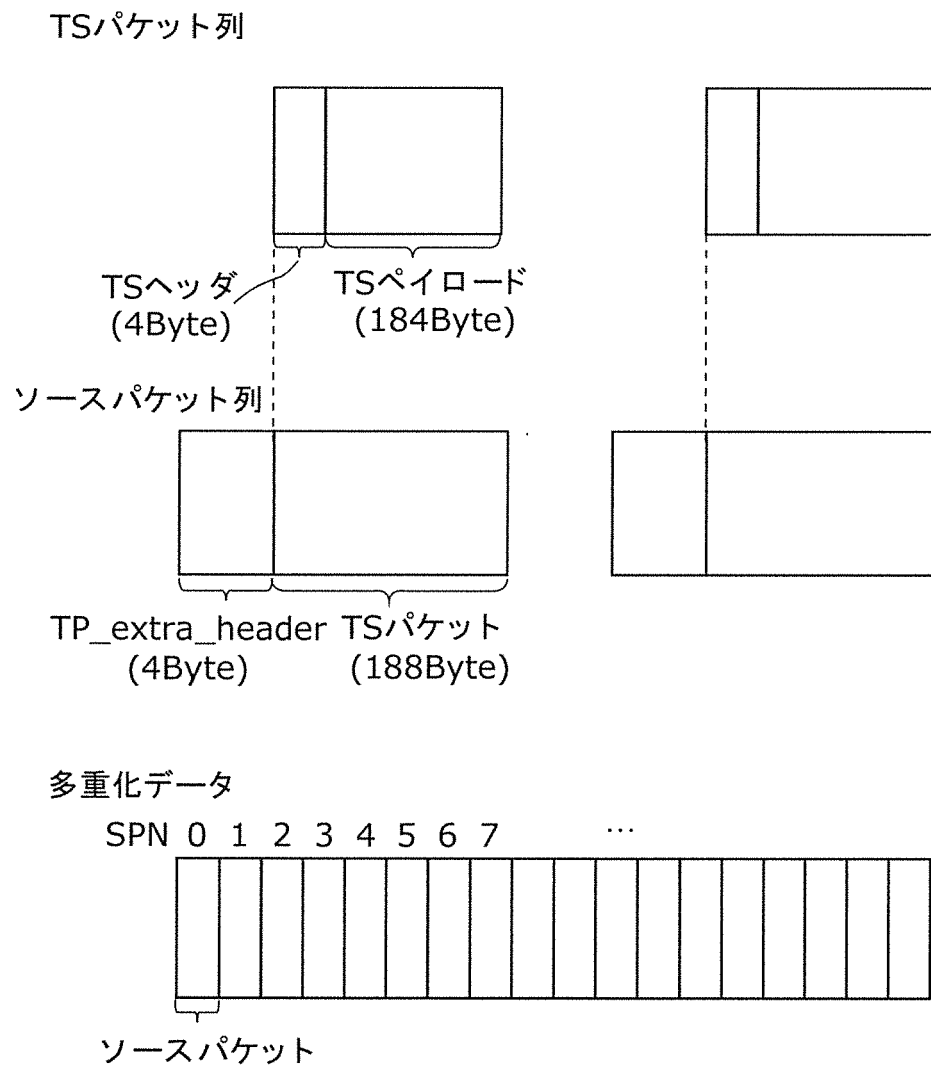
[図33]



[図34]

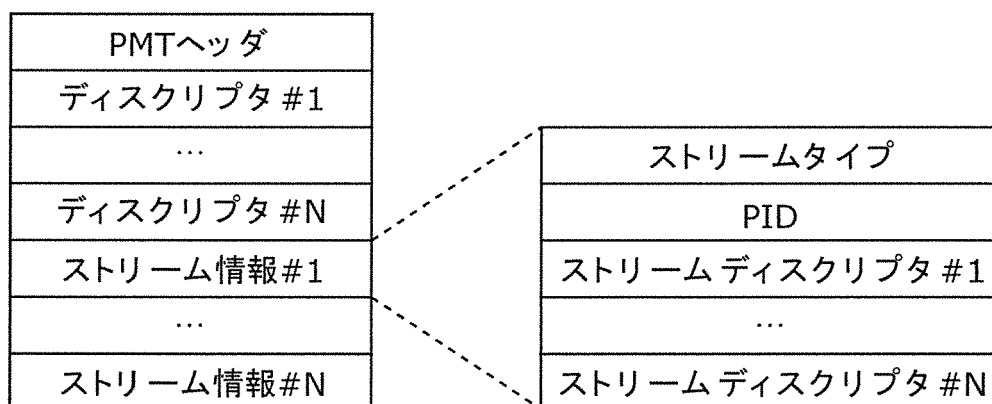


[図35]

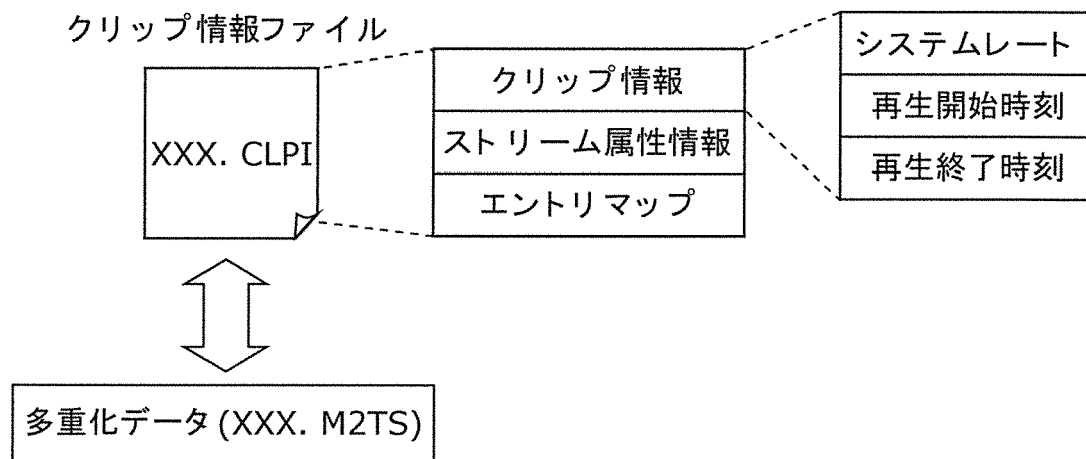


[図36]

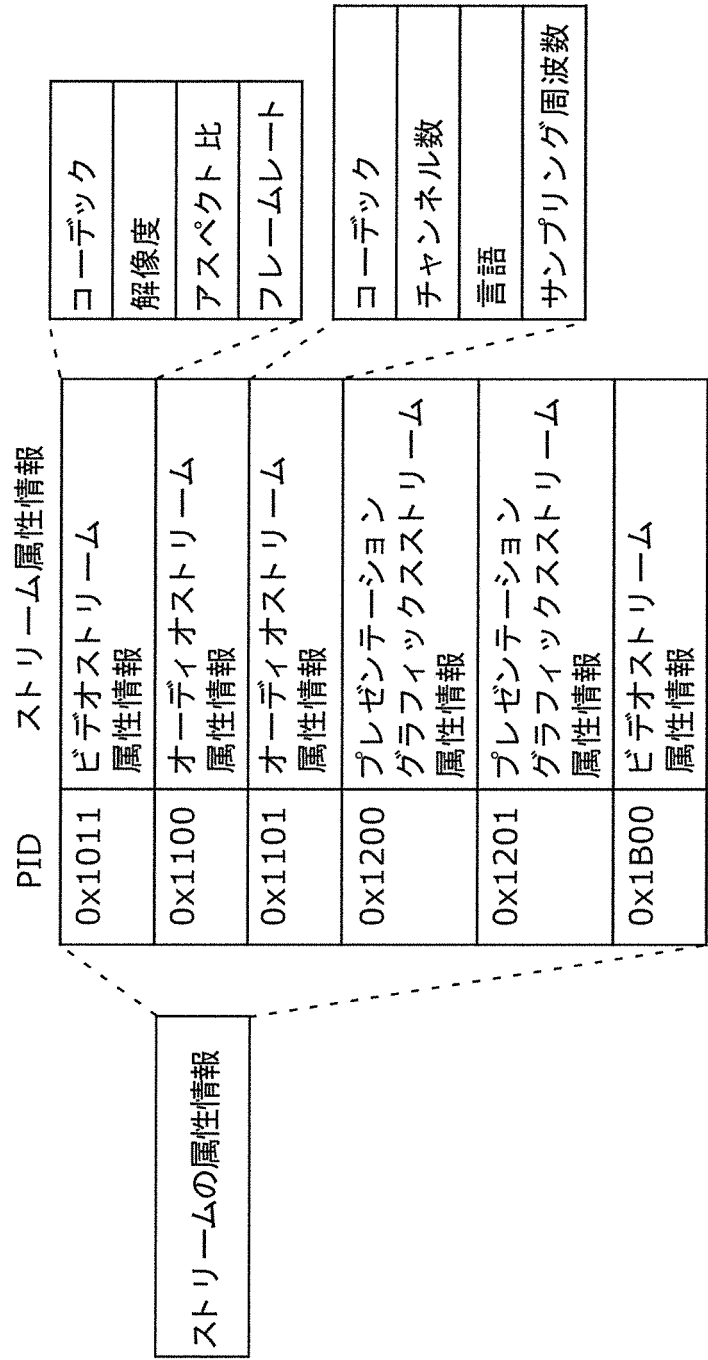
PMTのデータ構造



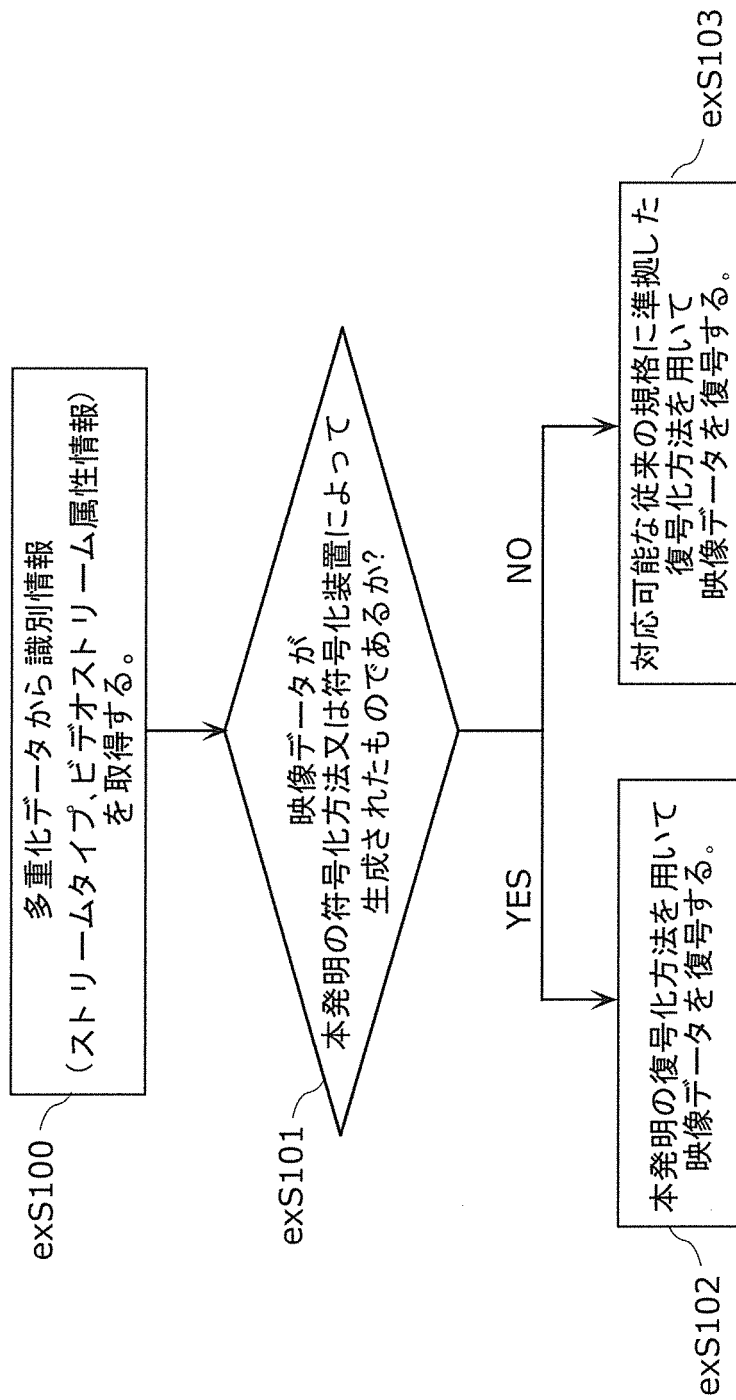
[図37]



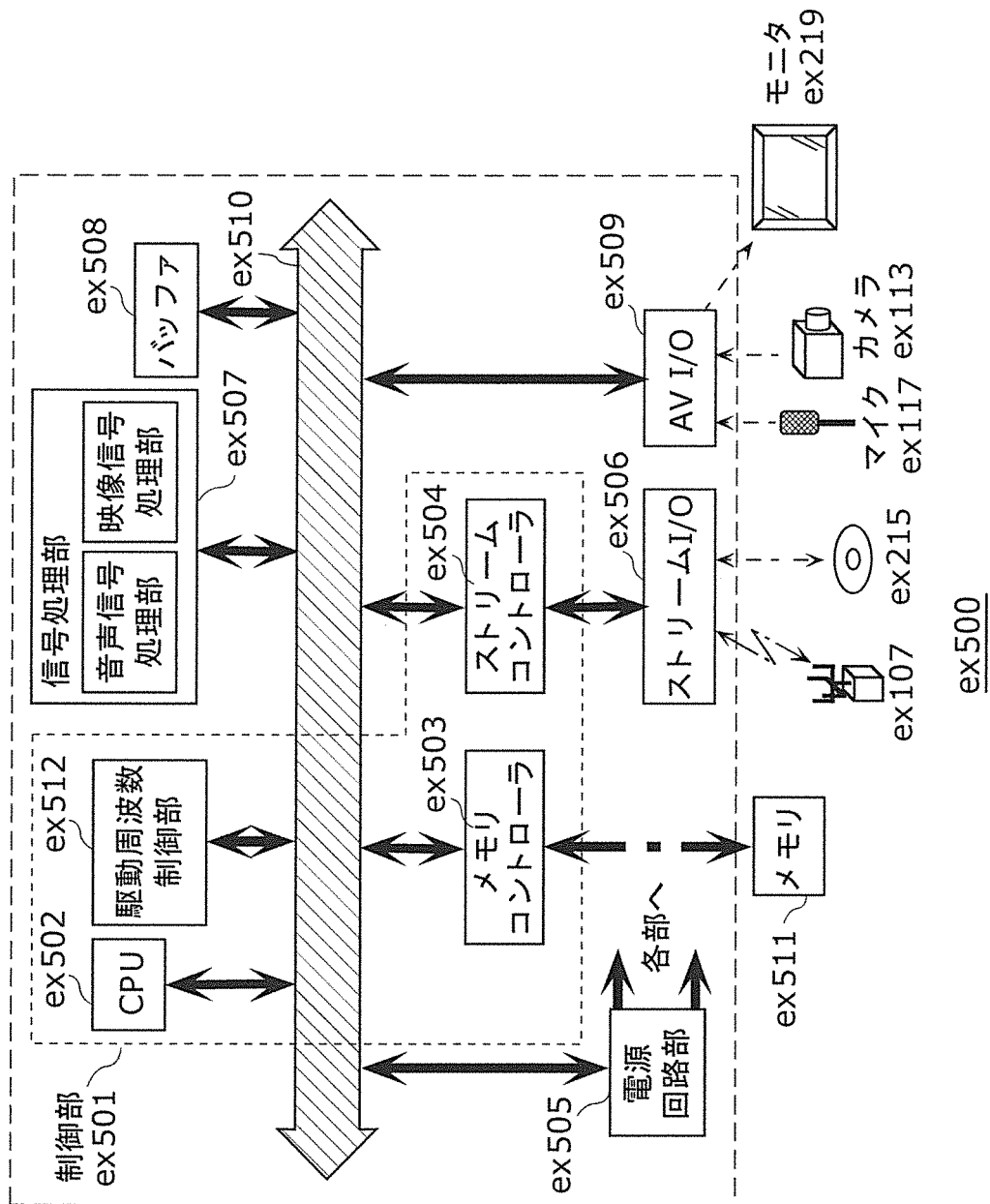
[図38]



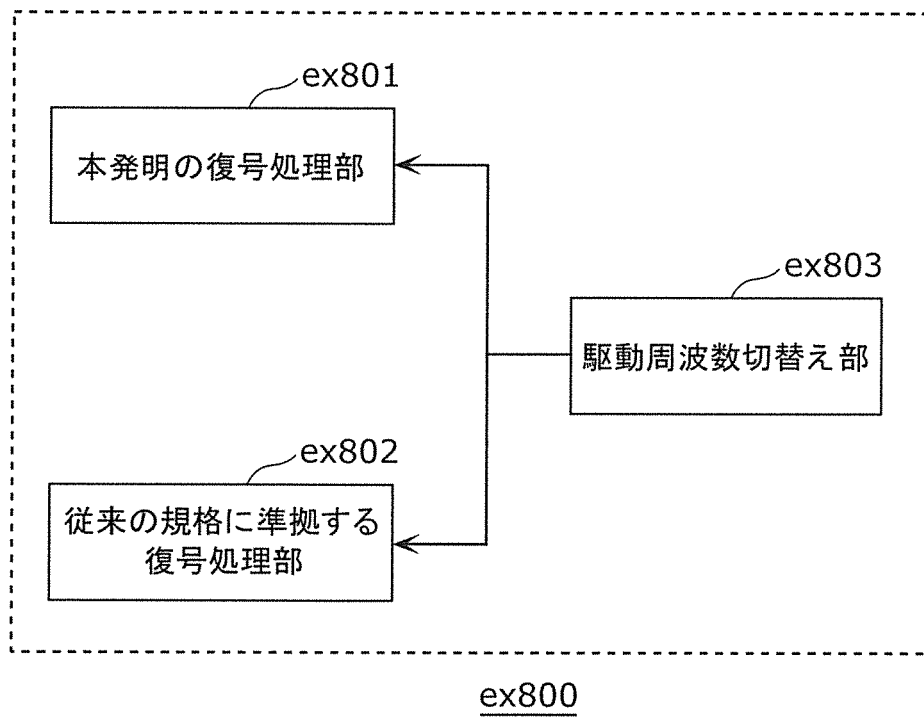
[図39]



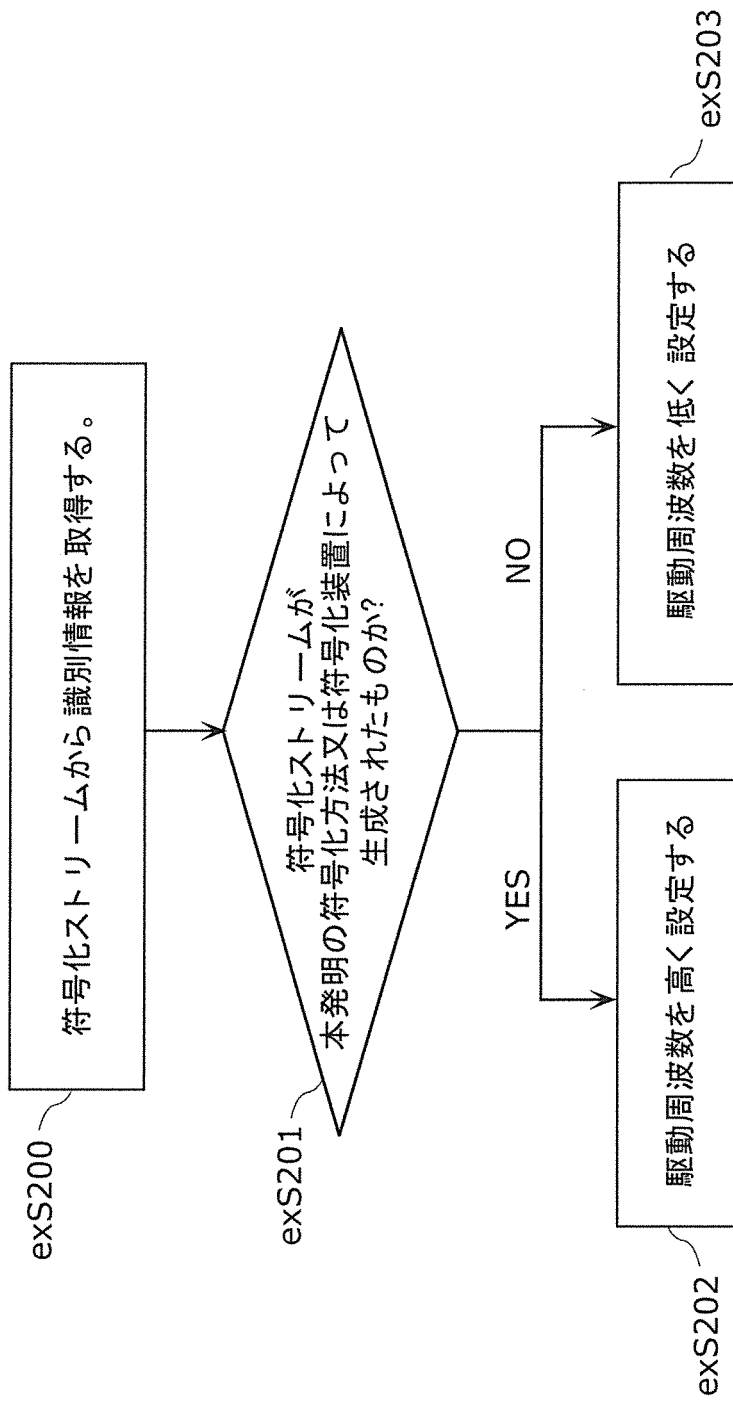
[図40]



[図41]



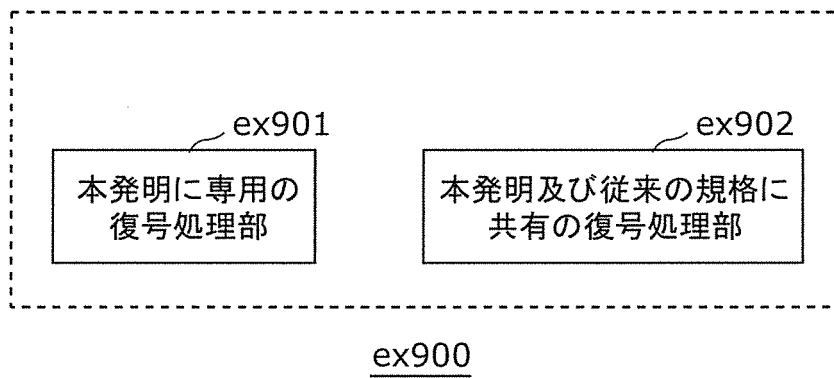
[図42]



[図43]

| 対応規格      | 駆動周波数  |
|-----------|--------|
| MPEG4.AVC | 500MHz |
| MPEG2     | 350MHz |
| ...       | ...    |

[図44A]



[図44B]

