

FIG. 1

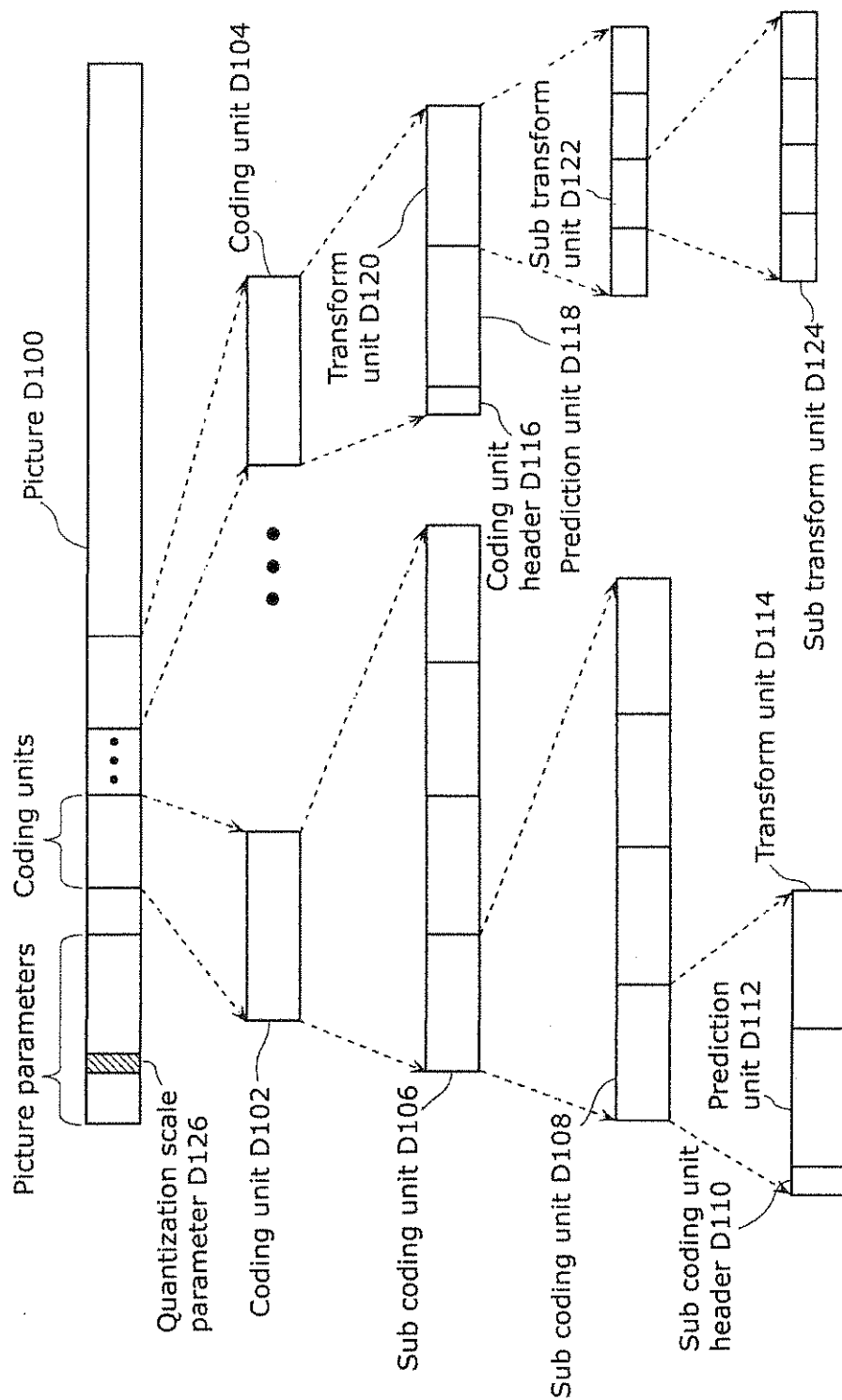


FIG. 2

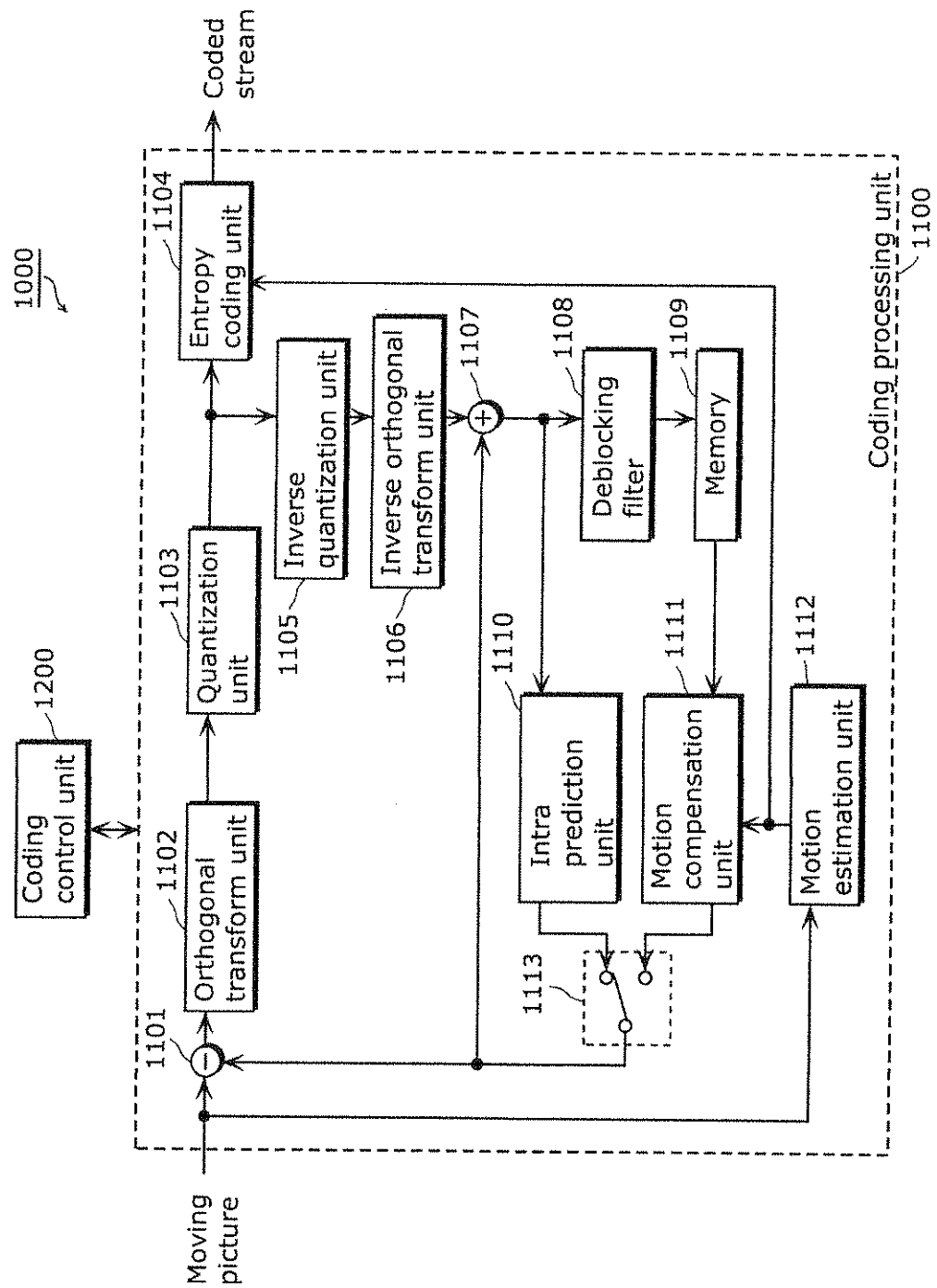


FIG. 3

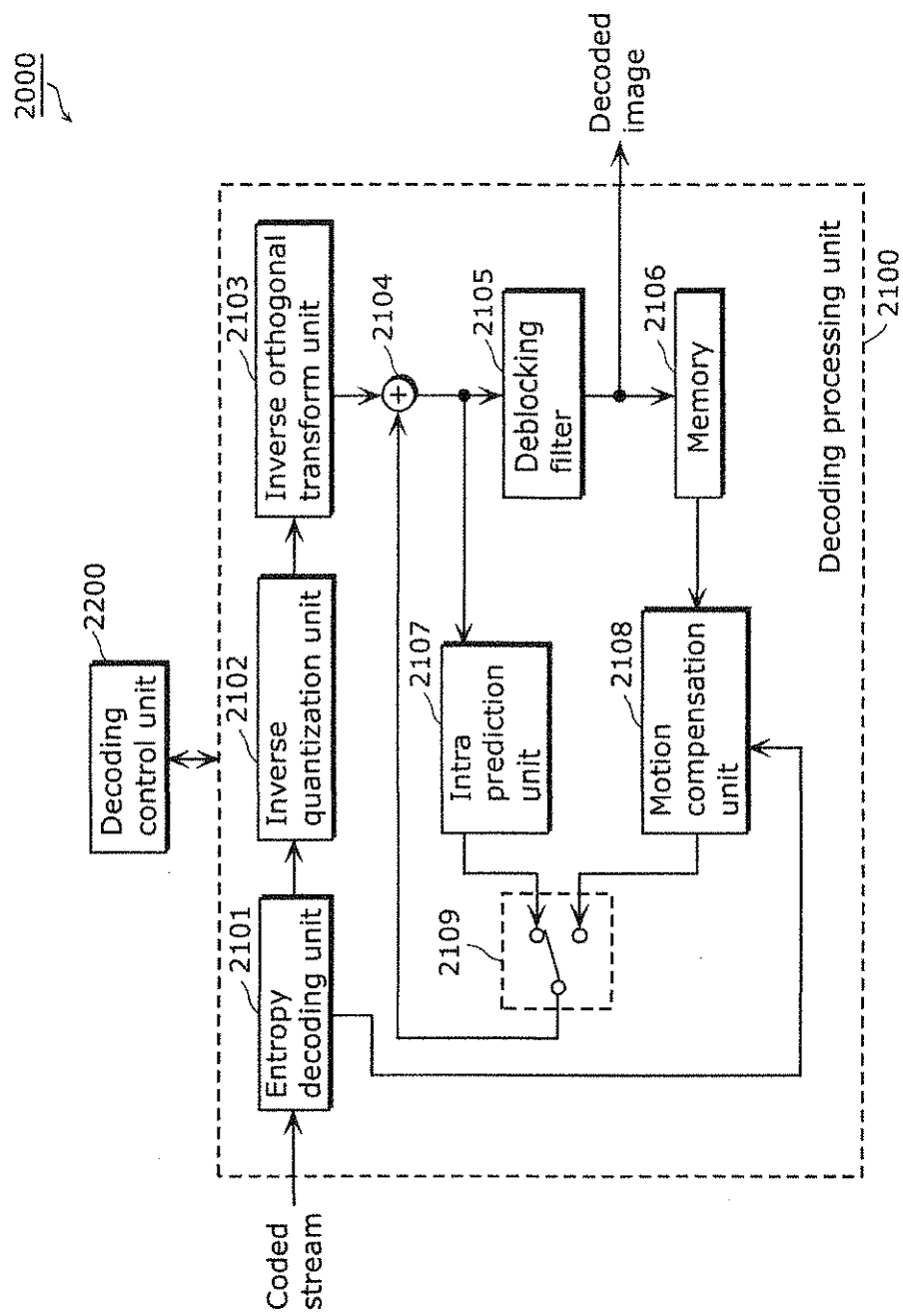


FIG. 4

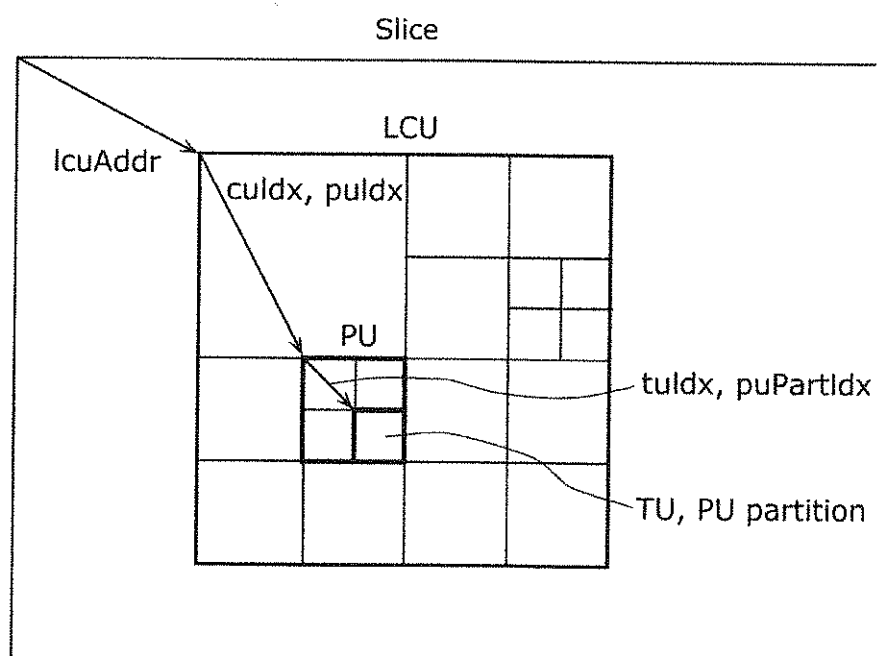


FIG. 5

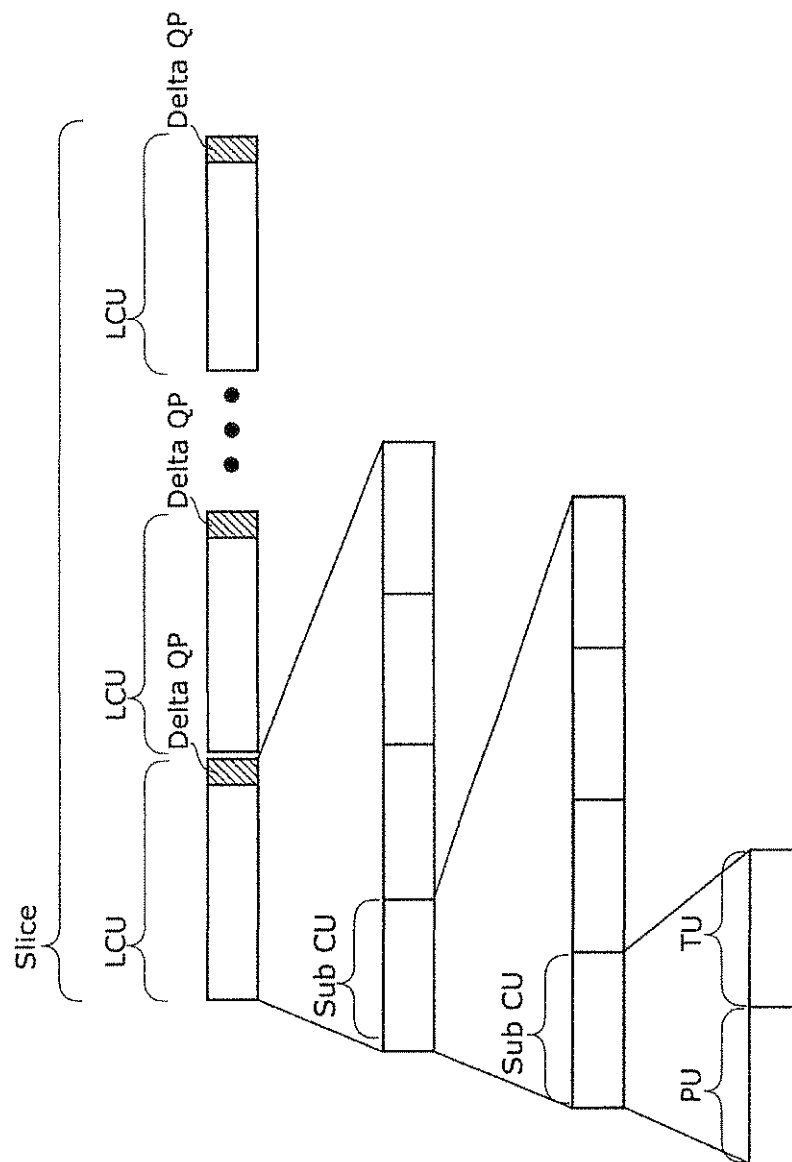


FIG. 6A

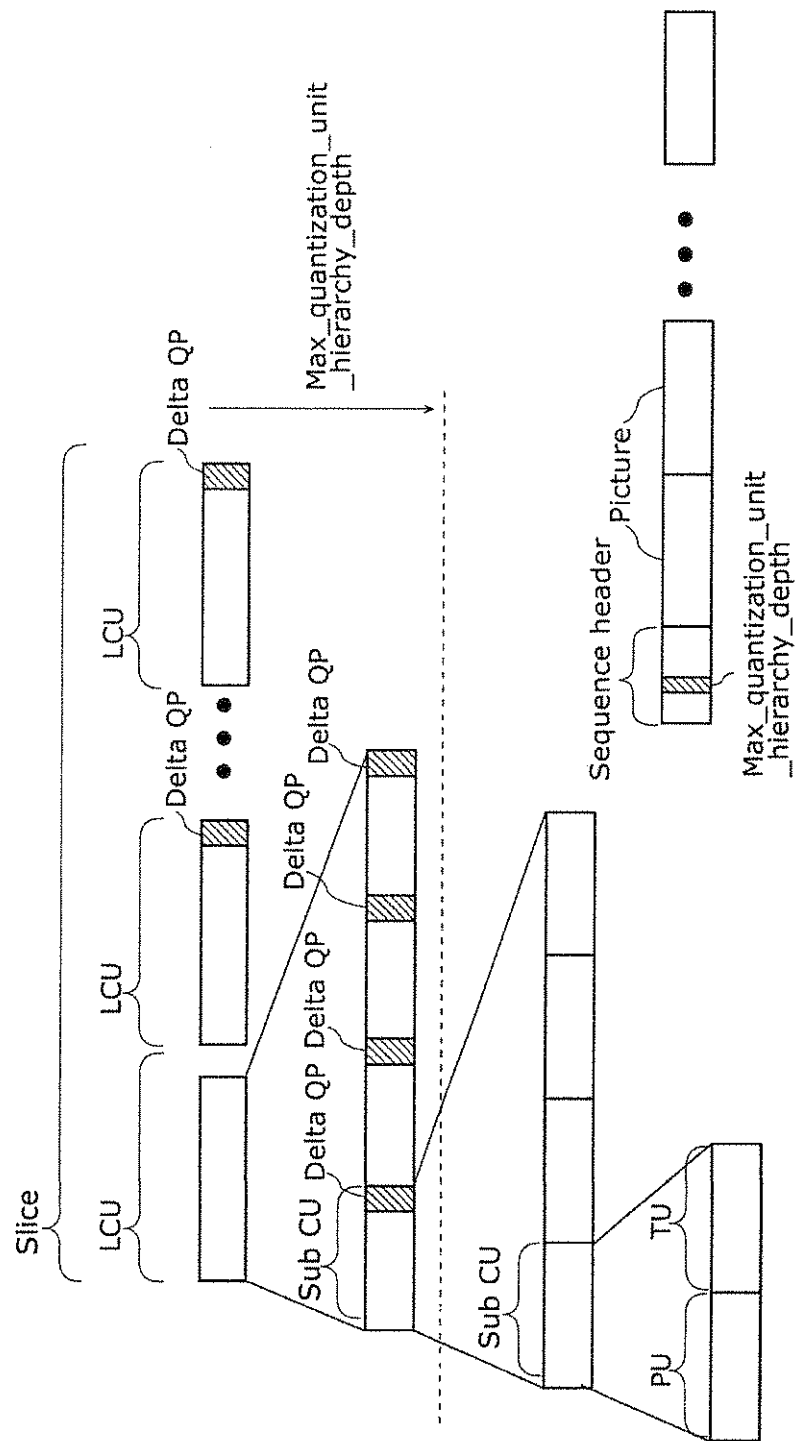


FIG. 6B

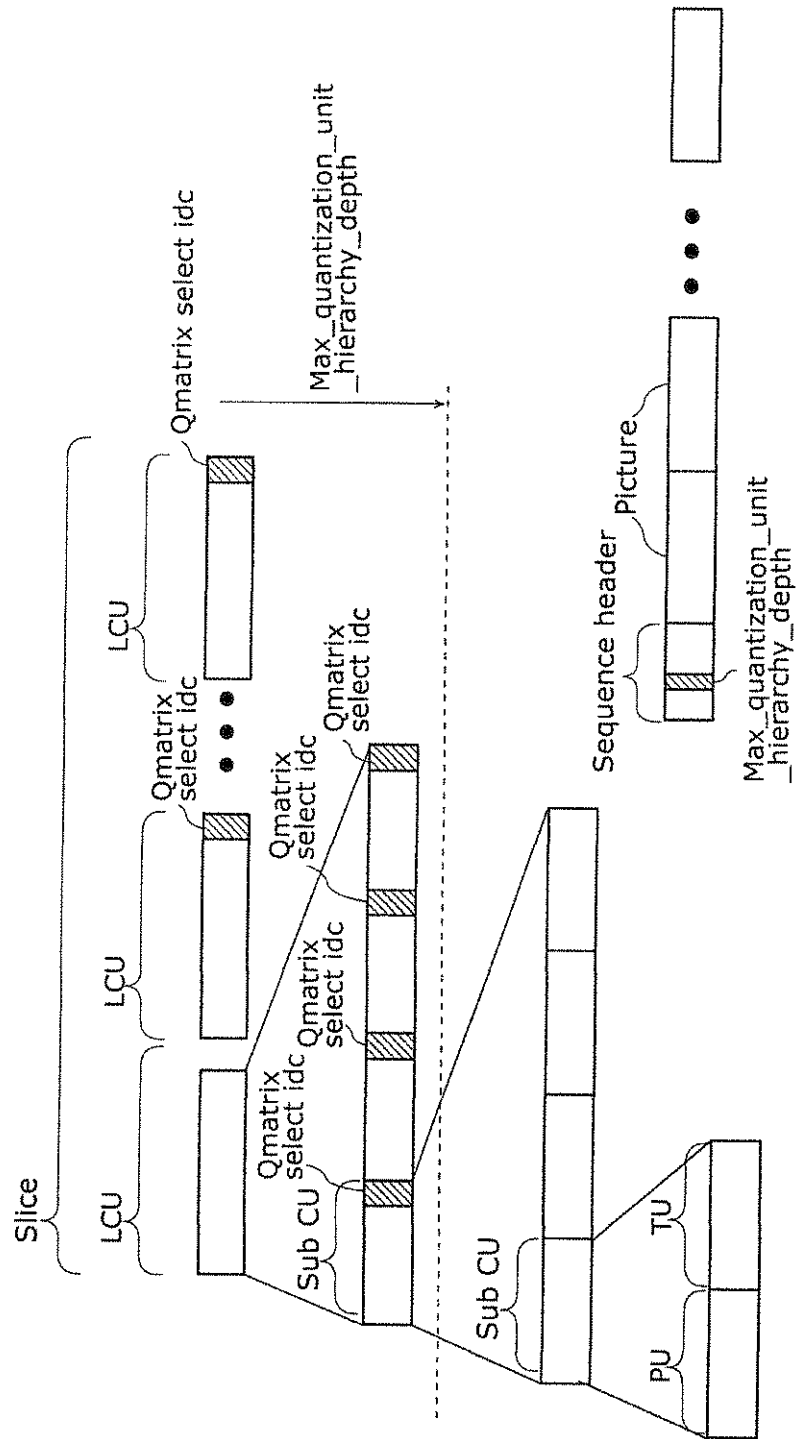


FIG. 6C

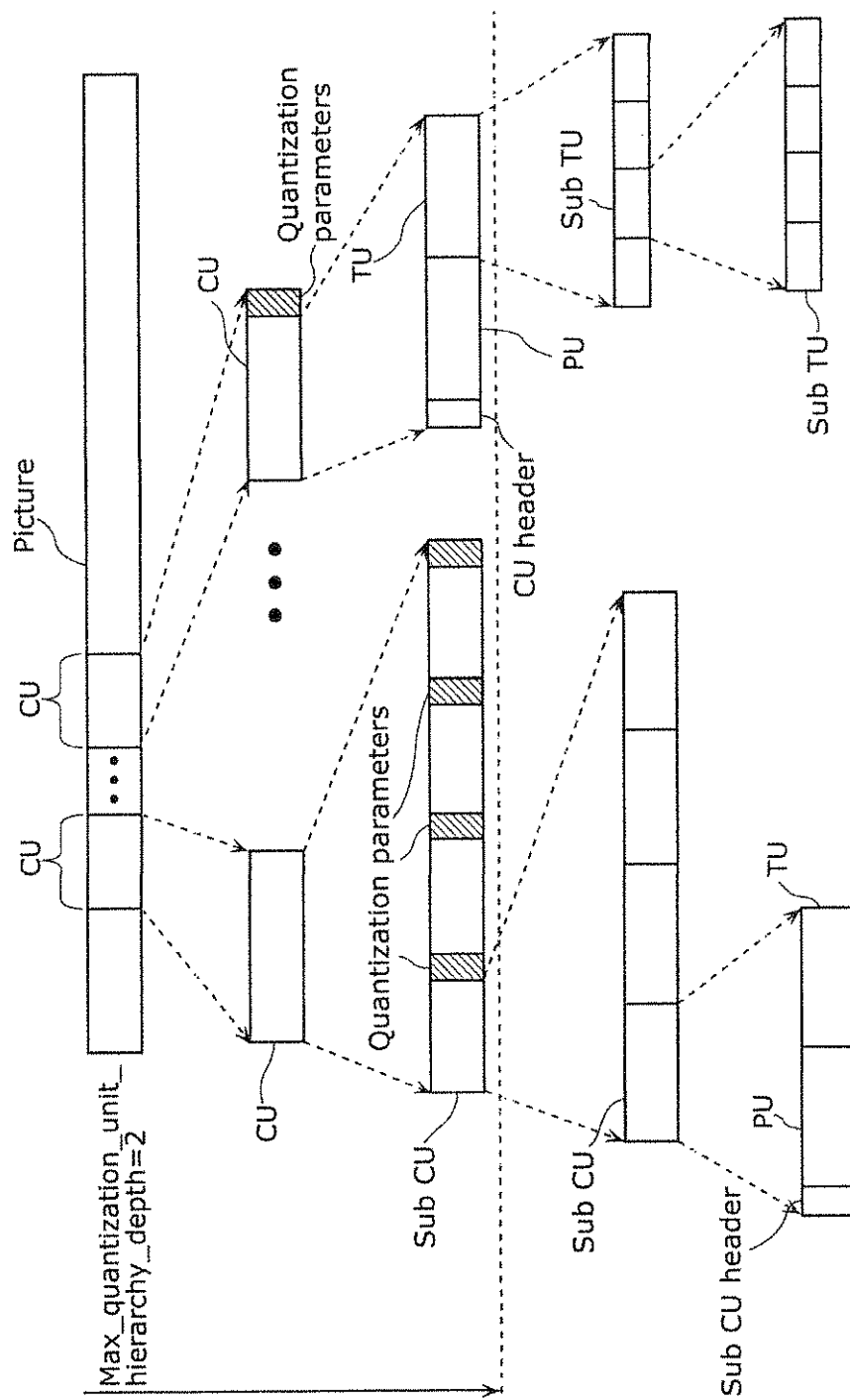


FIG. 7

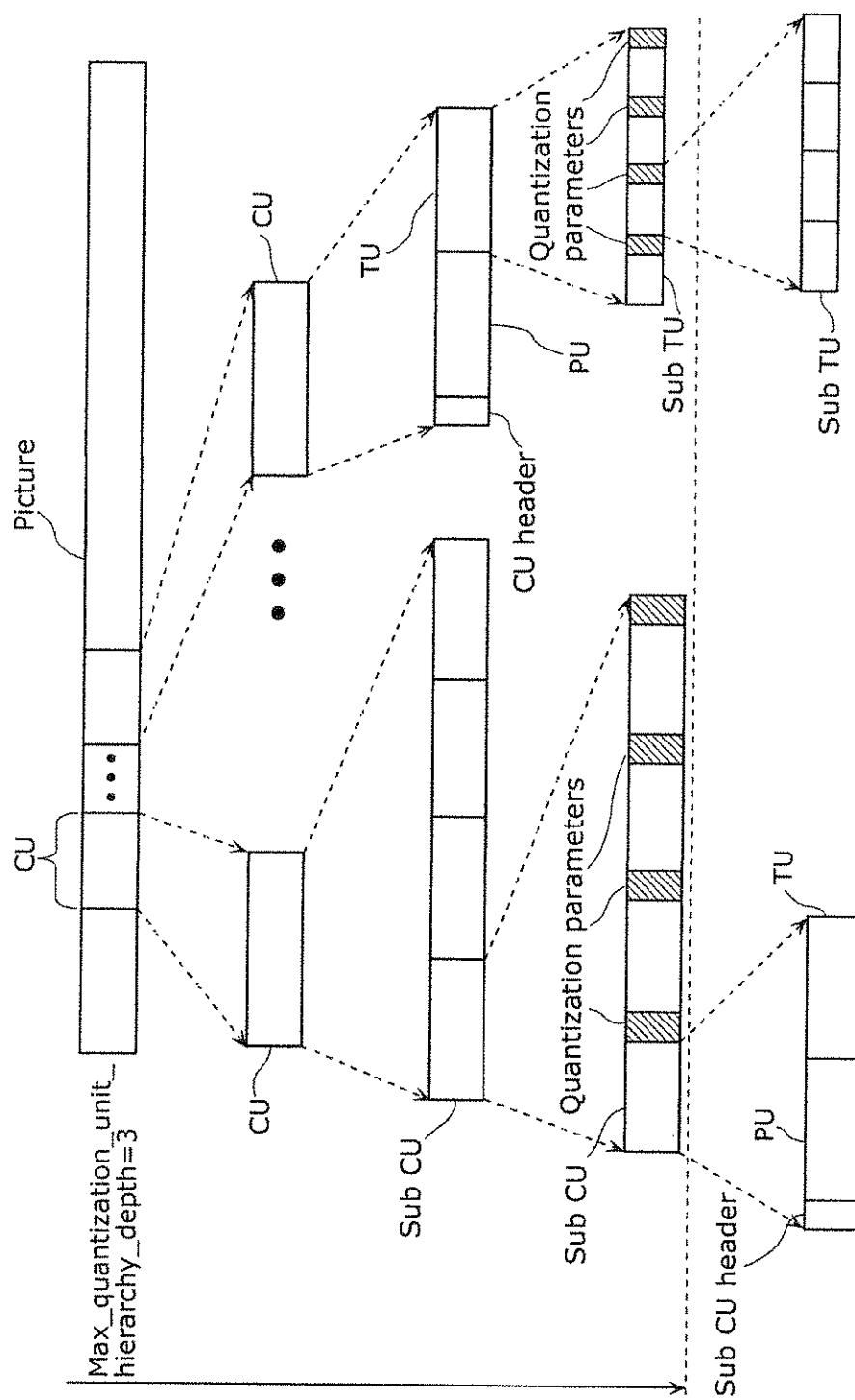


FIG. 8A

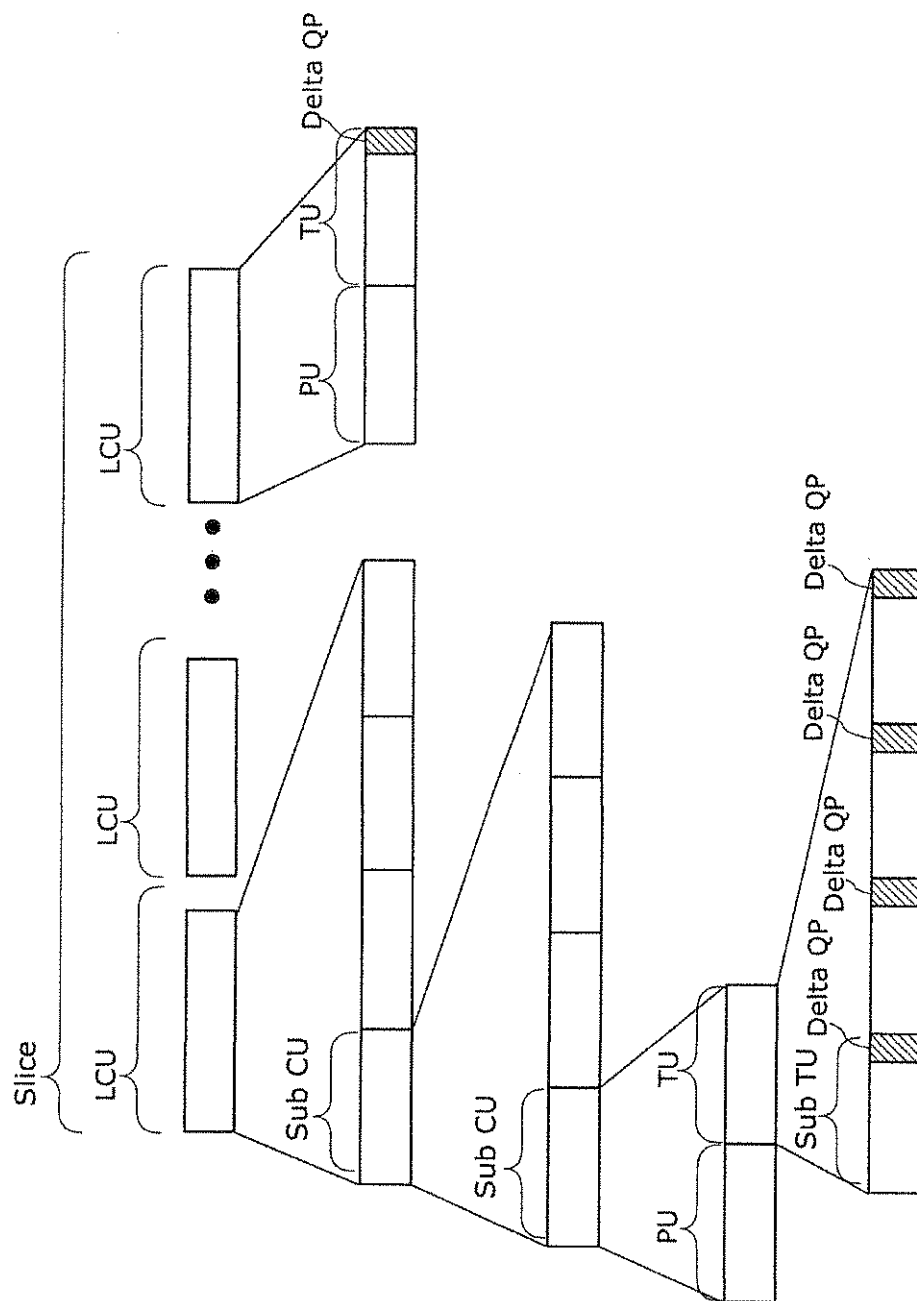


FIG. 9A

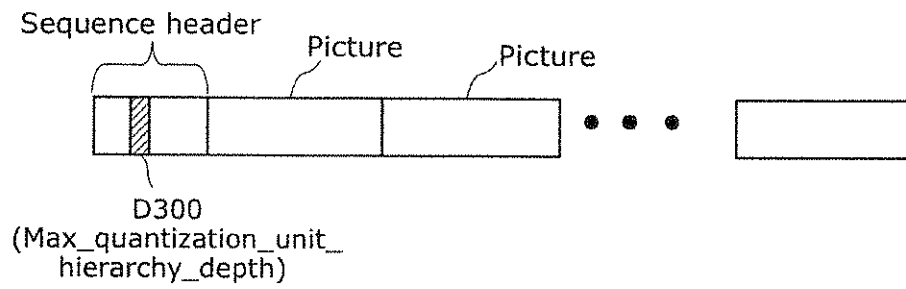


FIG. 9B

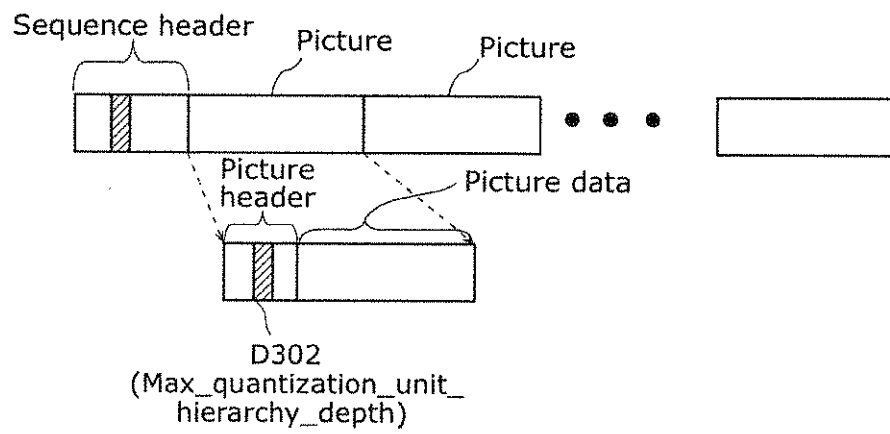


FIG. 10A

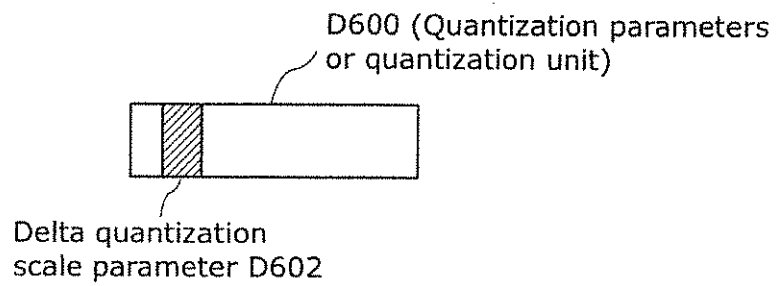


FIG. 10B

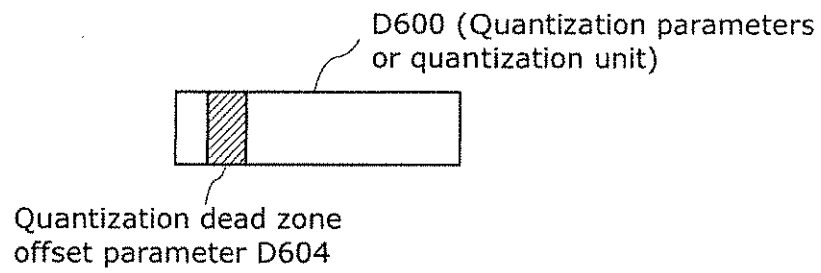


FIG. 10C

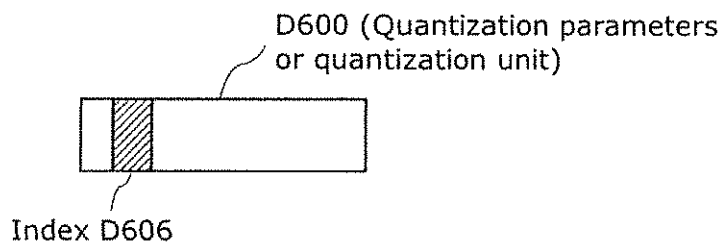


FIG. 10D

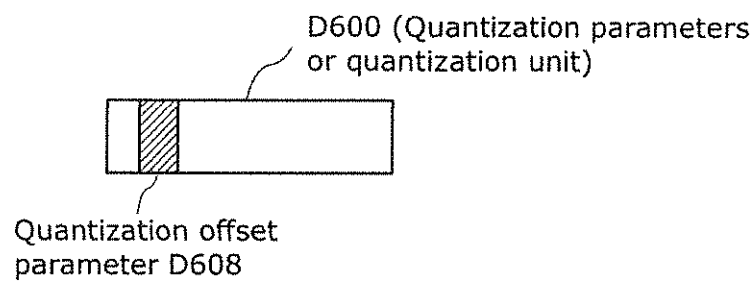


FIG. 11

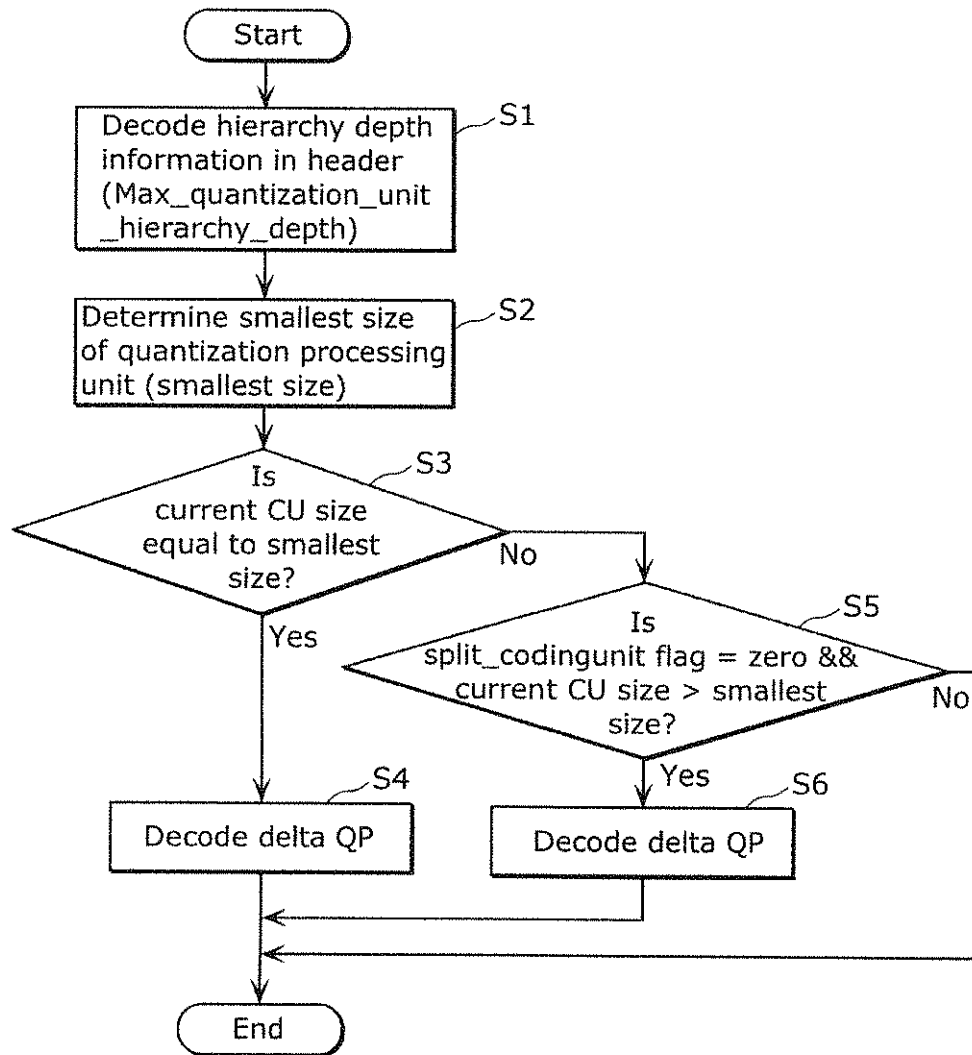


FIG. 12

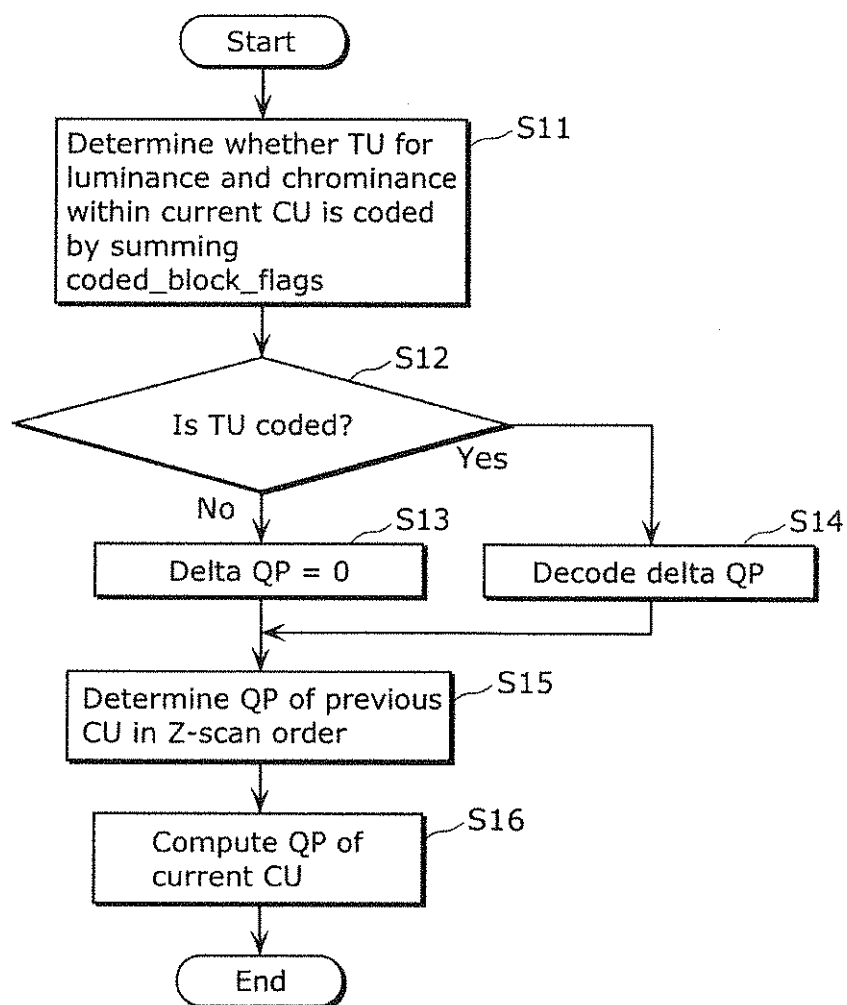


FIG. 13

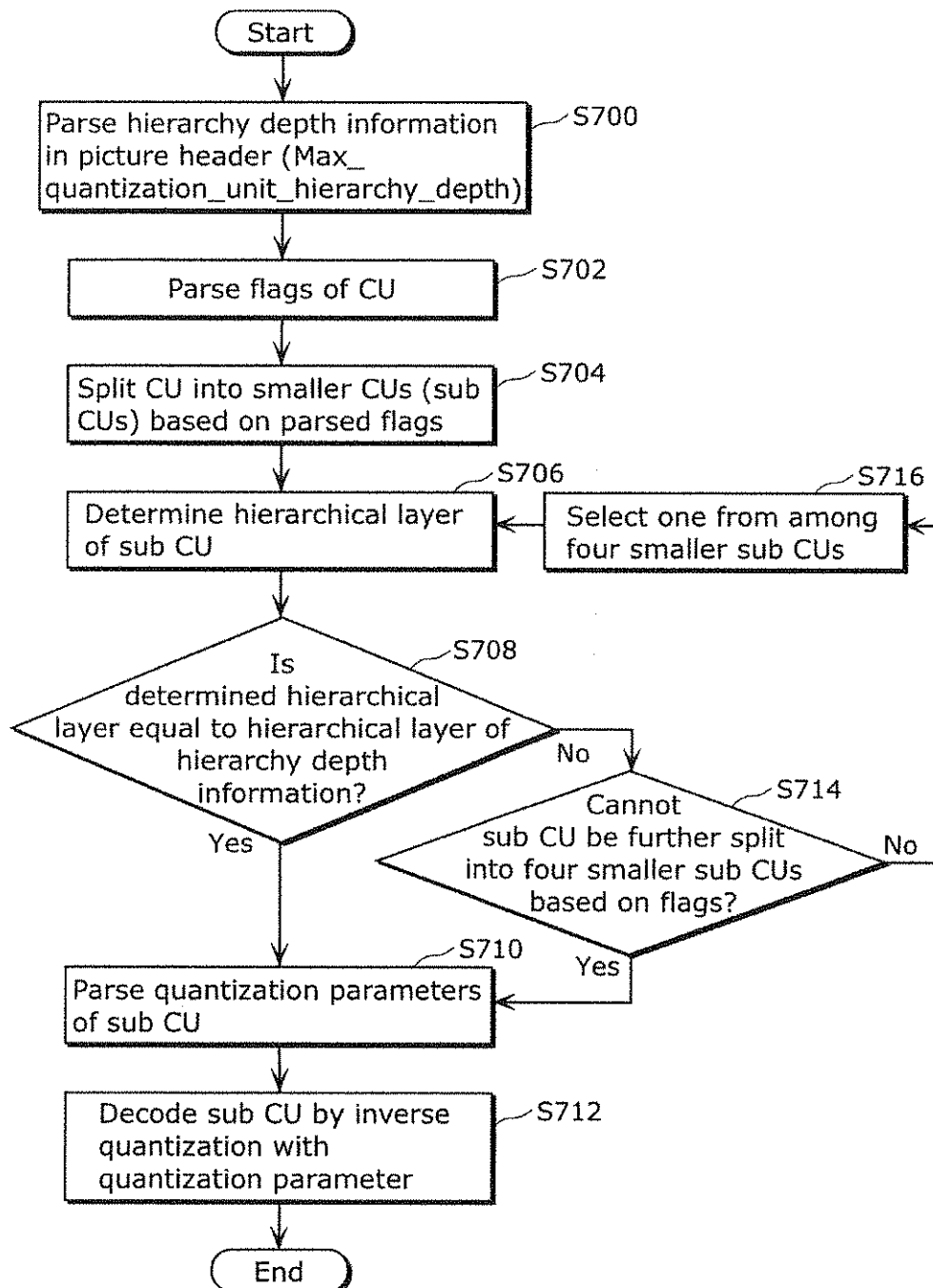


FIG. 14

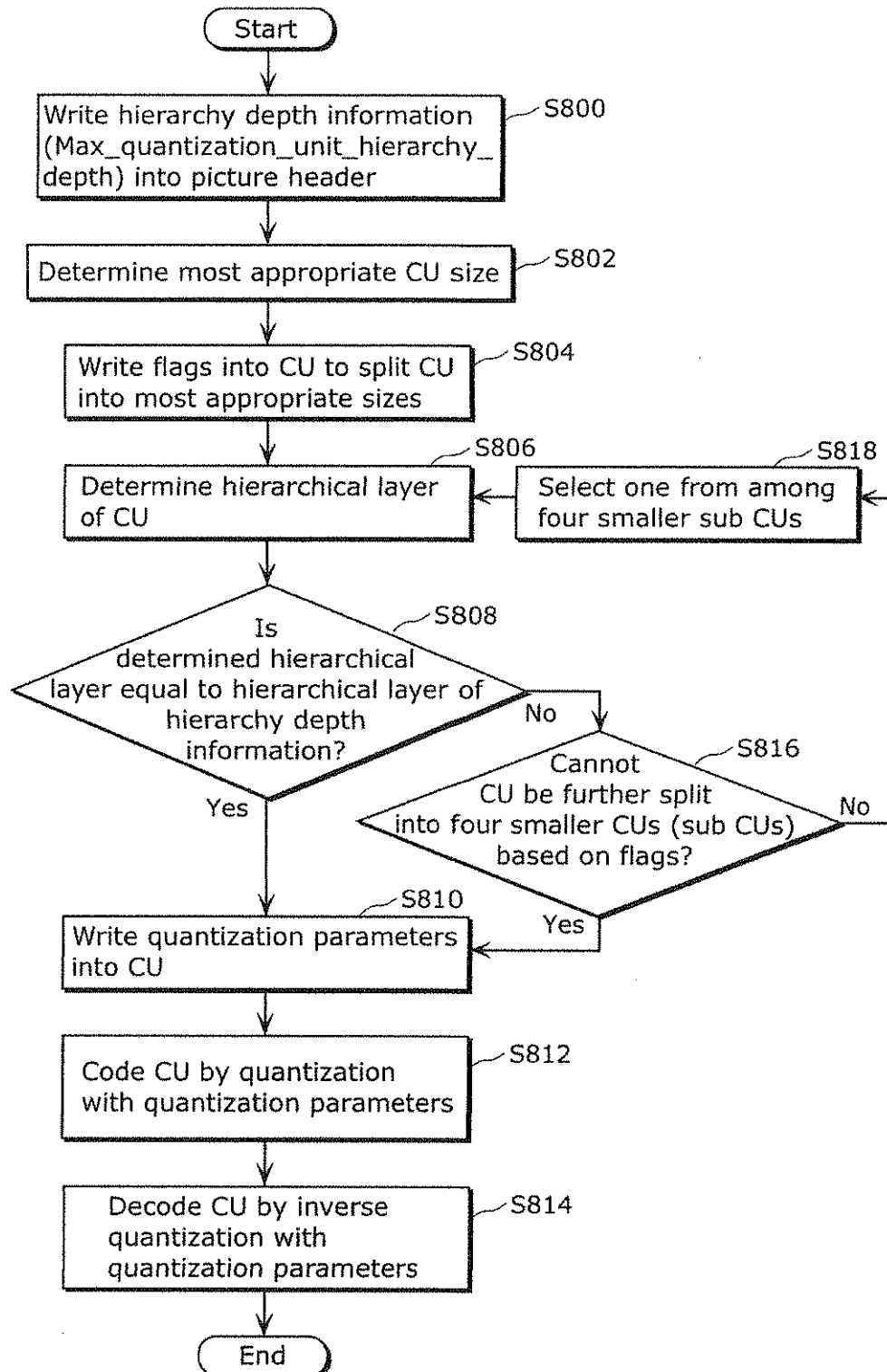


FIG. 15A

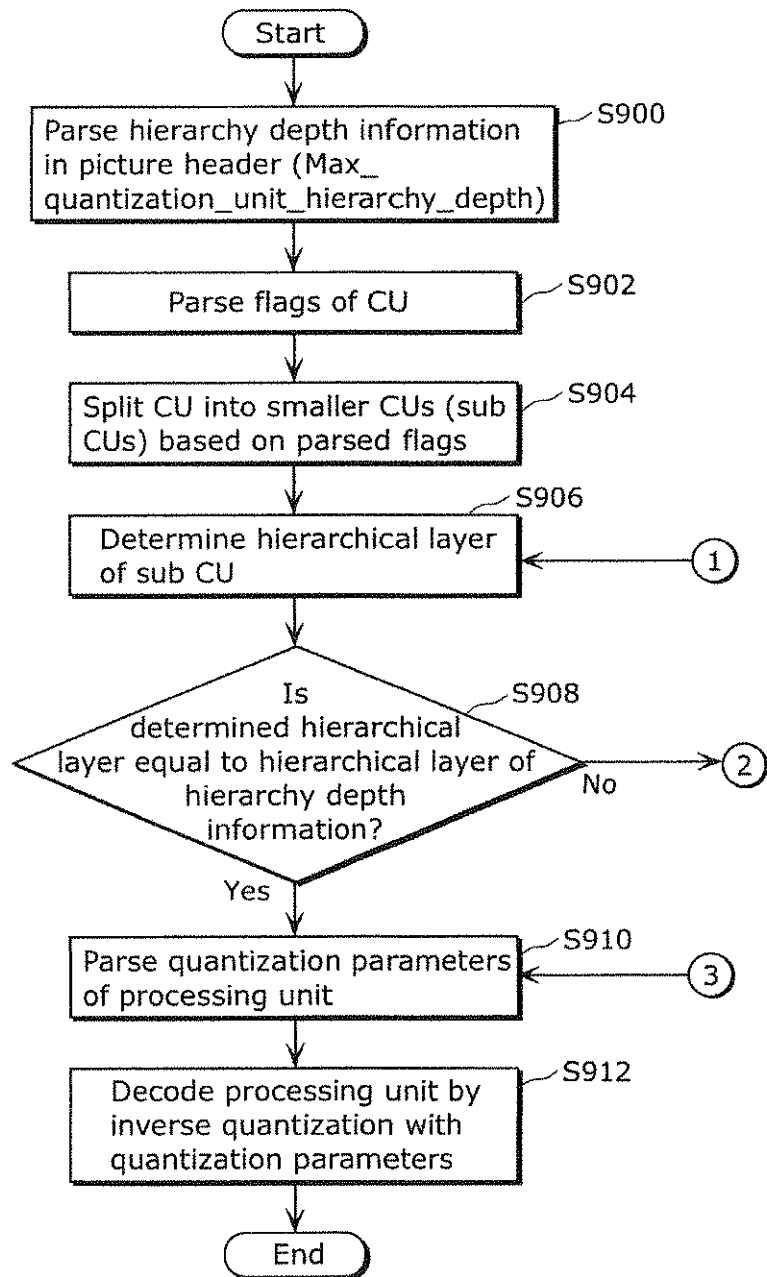


FIG. 15B

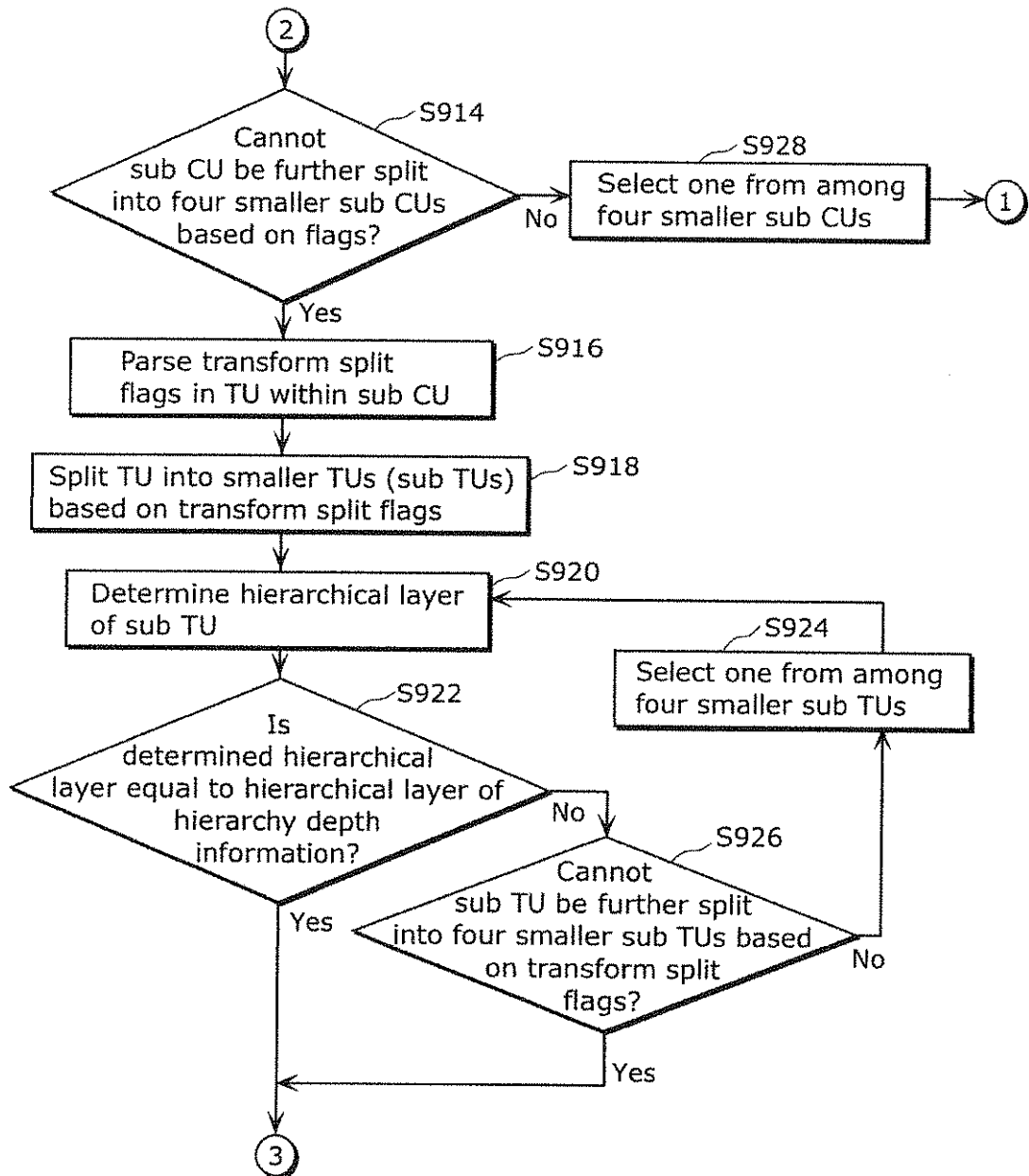


FIG. 16A

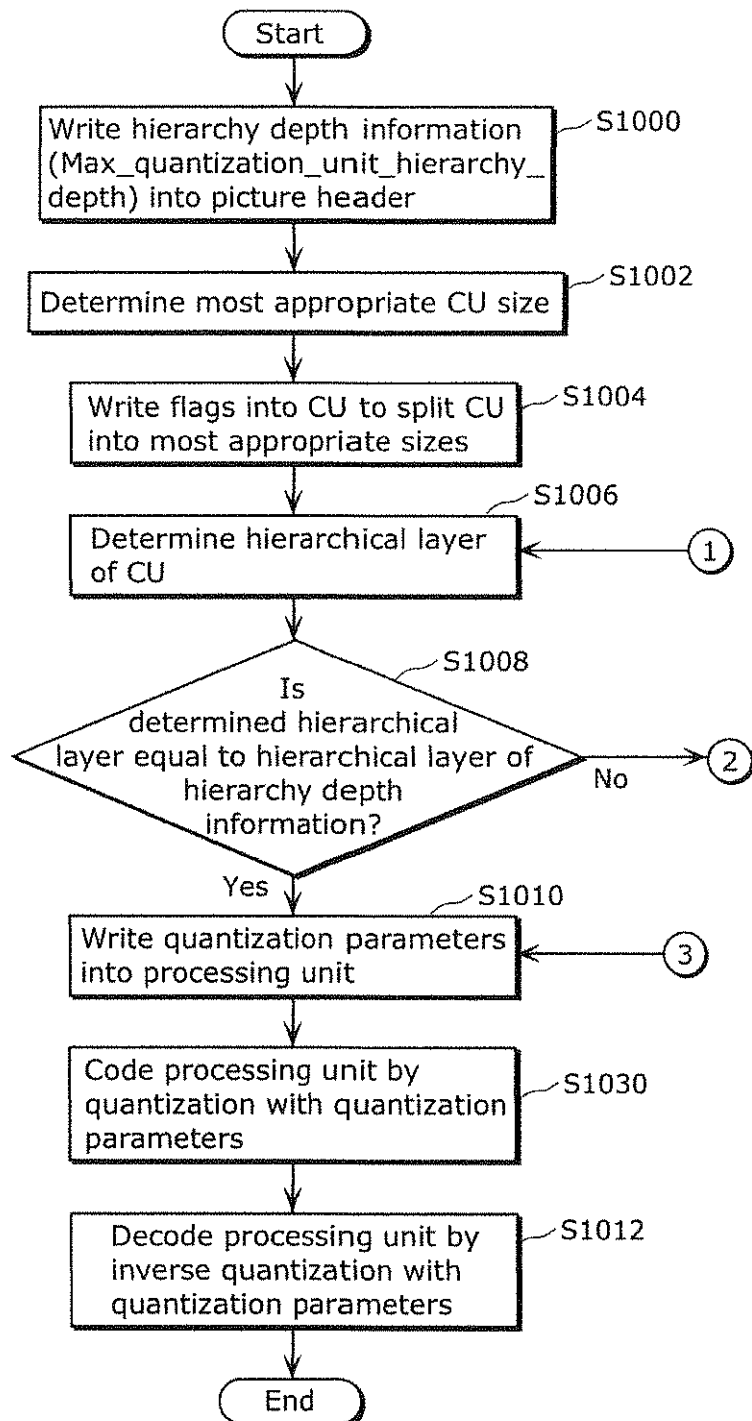


FIG. 16B

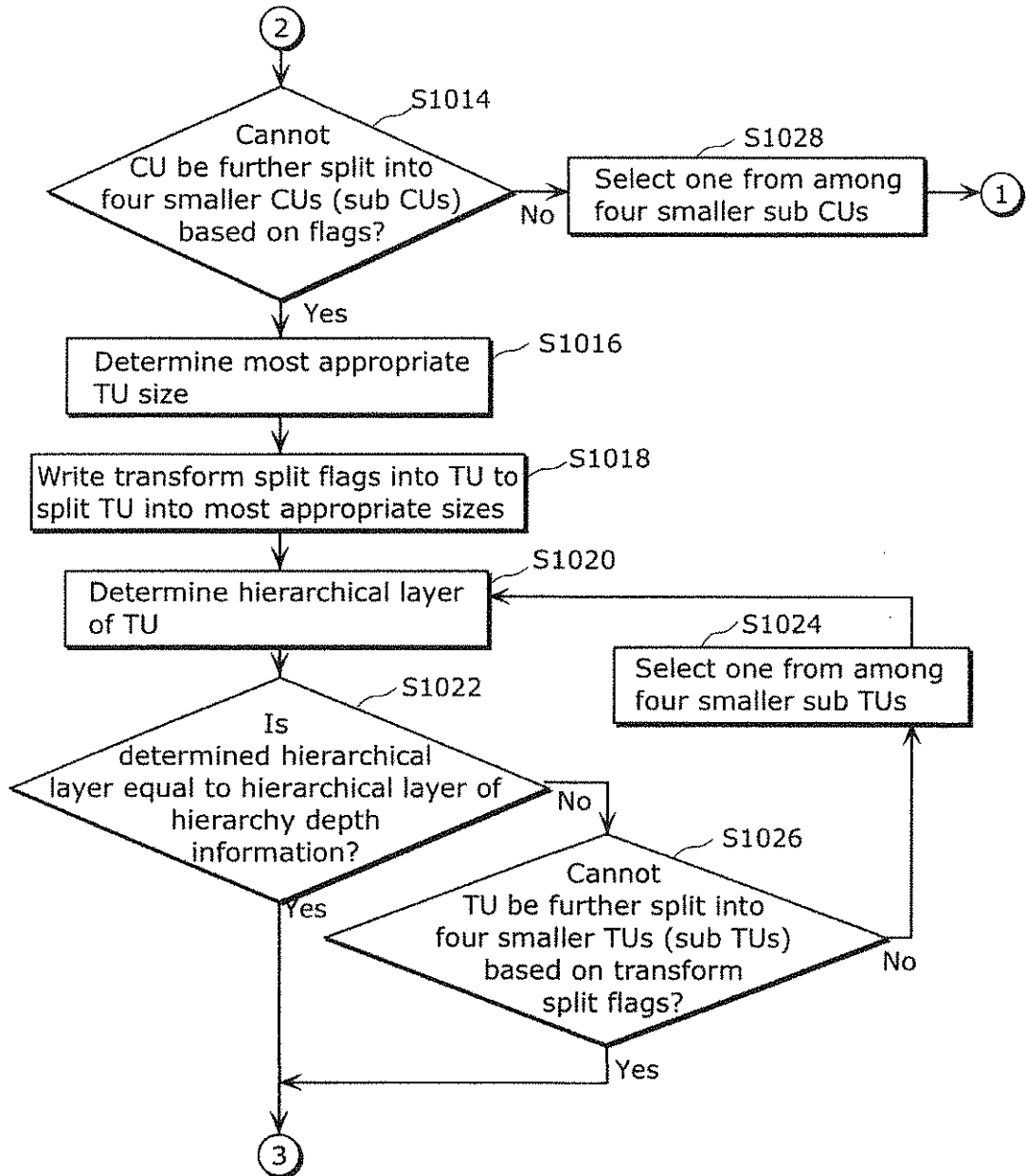


FIG. 17A

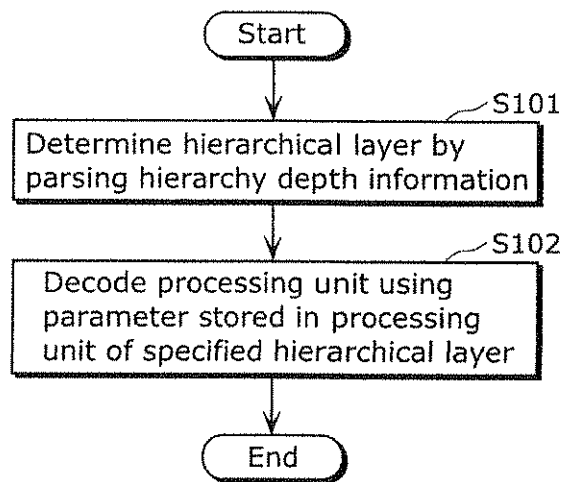


FIG. 17B

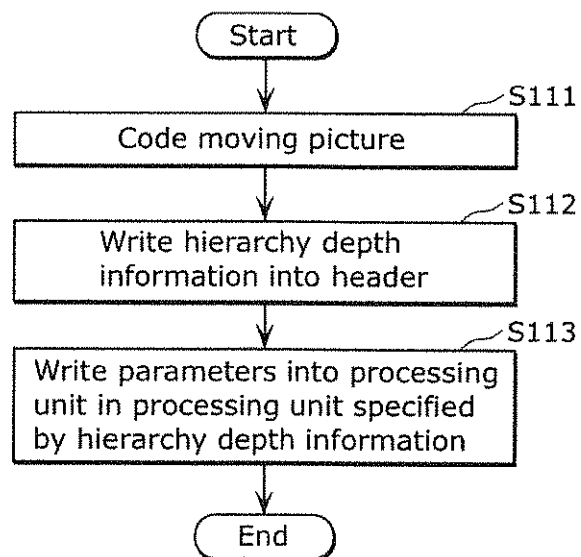


FIG. 18A

Sequence header

```
seq_parameter_set_rbsp( ) {
    profile_idc
    reserved_zero_8bits /* equal to 0 */
    level_idc
    seq_parameter_set_id
    bit_depth_luma_minus8
    bit_depth_chroma_minus8
    increased_bit_depth_luma
    line_bit_depth_chroma
    log2_max_frame_num_minus4
    log2_max_pic_order_cnt_lsb_minus4
    max_num_ref_frames
    gaps_in_frame_num_value_allowed_flag
    log2_min_coding_unit_size_minus3
    max_coding_unit_hierarchy_depth
    log2_min_transform_unit_size_minus2
    max_transform_unit_hierarchy_depth
    pic_width_in_luma_samples
    pic_height_in_luma_samples
    numExtraFilters
    for(i=0; i< numExtraFilters; i++){
        log2_filterCoeffPrecision
        halfNumTap
        for(j=0; j<(3*halfNumTap); j++){
            filterCoef[i][j]
        }
    }
    rbsp_trailing_bits( )
}
```

FIG. 18B

Picture header

```
pic_parameter_set_rbsp( ){  
    pic_parameter_set_id  
    seq_parameter_set_id  
    entropy_coding_mode_flag  
    num_ref_idx_l0_default_active_minus1  
    num_ref_idx_l1_default_active_minus1  
    pic_init_qp_minus26 /* relative to 26 */  
    constrained_intra_pred_flag  
    for(i=0; i<15; i++){  
        numAllowedFilters[i]  
        for(j=0; j<numAllowedFilters[i]; j++){  
            filtIdx[i][j]  
        }  
    }  
    rbsp_trailing_bits()  
}
```

d1

FIG. 18C

Slice header

```

slice_header( ) {
    first_lctb_in_slice
    slice_type
    pic_parameter_set_id
    frame_num
    if( IdrPicFlag )
        idr_pic_id
    pic_order_cnt_lsb
    if( slice_type == P || slice_type == B ) {
        num_ref_idx_active_override_flag
        if( num_ref_idx_active_override_flag ) {
            num_ref_idx_l0_active_minus1
            if( slice_type == B )
                num_ref_idx_l1_active_minus1
        }
    }
    if( nal_ref_idc != 0 )
        dec_ref_pic_marking( )
    if( entropy_coding_mode_flag && slice_type != I )
        cabac_init_idc
    slice_qp_delta
    alt_param()
    if( slice_type == P || slice_type == B ) {
        mc_interpolation_idc
        mv_competition_flag
        if( mv_competition_flag ) {
            mv_competition_temporal_flag
        }
    }
    if( slice_type == B && mv_competition_flag )
        collocated_from_l0_flag

    sifo_param( )
    if( entropy_coding_mode_flag == 3 )
        parallel_v2v_header( )
    edge_based_prediction_flag
    if( edge_prediction_ipd_flag == 1 )
        threshold_edge
}

```

d2

d3

FIG. 19A

```

CU
coding_unit( x0,y0, currCodingUnitSize ) {
    if( x0+currCodingUnitSize < PicWidthInSamples_L && y0+currCodingUnitSize <
        PicHeightInSamples_L && currCodingUnitSize > MinCodingUnitSize )
        split_coding_unit_flag
        if( ( split_coding_unit_flag && currCodingUnitSize == AlfMinCtrlCodingUnitSize ) ||
            ( !split_coding_unit_flag && currCodingUnitSize > AlfCtrlMinCodingUnitSize ) )
            alf_flag
            if( split_coding_unit_flag ) {
                splitCodingUnitSize = currCodingUnitSize > > 1
                x1 = x0 + splitCodingUnitSize
                y1 = y0 + splitCodingUnitSize
                coding_unit( x0, y0, splitCodingUnitSize )
                if( x1 < PicWidthInSamples_L )
                    coding_unit( x1, y0, splitCodingUnitSize )
                if( y1 < PicHeightInSamples_L )
                    coding_unit( x0, y1, splitCodingUnitSize )
                if( x1 < PicWidthInSamples_L && y1 < PicHeightInSamples_L )
                    coding_unit( x1, y1, splitCodingUnitSize )
            }
            else {
                prediction_unit( x0, y0, currCodingUnitSize )
                if( PredMode != MODE_SKIP ) { PredMode = MODE_INTRA && planar_flag = 1 )
                    transform_unit( x0, y0, currCodingUnitSize )
            }
        }
    }
}

```

d4

d5

FIG. 19B

```

PU
prediction_unit( x0, y0, currPredUnitSize ) {
    if( slice_type != I )
        skip_flag
    if( mv_flag ) {
        if( mv_competition_flag ) {
            if( inter_pred_idc != Pred_L1 && NumMVPcand( L0 ) > 1 )
               .mvp_idx_l0
            if( inter_pred_idc != Pred_L0 && NumMVPcand( L1 ) > 1 )
               .mvp_idx_l1
        }
    }
    else {
        if( entropy_coding_mode_flag )
            mode_table_idx
        else {
            if( slice_type != I )
                pred_mode
        }
        if( PredMode == MODE_INTRA ) {
            planar_flag
            if( planar_flag ) {
                planar_delta_u = getPlanarDelta()
                planar_delta_v = present_flag
                if( planar_delta_u == present_flag ) {
                    planar_delta_u = getPlanarDelta()
                    planar_delta_v = getPlanarDelta()
                }
            }
            else {
                if( entropy_coding_mode_flag )
                    combined_intra_pred_flag
                for( i = 0; i < ( intra_split_flag ? 4 : 1 ); i++ ) {
                    prev_intra_luma_pred_flag
                    if( !prev_intra_luma_pred_flag )
                        rem_intra_luma_pred_mode
                }
            }
            if( chroma_format_idc != 0 )
                intra_chroma_pred_mode
        }
    }
    else if( PredMode == MODE_INTER ) {
        if( entropy_coding_mode_flag )
            inter_partitioning_idc
        for( i = 0; i < NumPartitions( inter_partitioning_idc ); i++ ) {
            if( NumMergeCandidates > 0 ) {
                merge_flag[i]
                if( merge_flag[i] && NumMergeCandidates > 1 )
                    merge_left_flag[i]
            }
            if( !merge_flag[i] ) {
                if( slice_type == B )
                    inter_pred_idc[i]
                if( inter_pred_idc[i] != Pred_L1 ) {
                    if( entropy_coding_mode_flag ) {
                        if( num_ref_idx_l0_active_minus1 > 0 )
                            ref_idx_l0[i]
                        mvres_l0[i]
                    }
                    else {
                        ref_idx_mvres_l0[i]
                    }
                }
                mvd_l0[i][0]
                mvd_l0[i][1]
                if( mv_competition_flag && NumMVPcand( L0, i ) > 1 )
                   .mvp_idx_l0[i]
            }
            if( inter_pred_idc[i] != Pred_L0 ) {
                if( entropy_coding_mode_flag ) {
                    if( num_ref_idx_l1_active_minus1 > 0 )
                        ref_idx_l1[i]
                        mvres_l1[i]
                    }
                    else {
                        ref_idx_mvres_l1[i]
                    }
                }
                mvd_l1[i][0]
                mvd_l1[i][1]
                if( mv_competition_flag && NumMVPcand( L1, i ) > 1 )
                   .mvp_idx_l1[i]
            }
        }
    }
    else if( PredMode == MODE_DIRECT ) {
        if( slice_type == B )
            inter_pred_idc
        if( mv_competition_flag ) {
            if( inter_pred_idc != Pred_L1 && NumMVPcand( L0 ) > 1 )
               .mvp_idx_l0
            if( inter_pred_idc != Pred_L0 && NumMVPcand( L1 ) > 1 )
               .mvp_idx_l1
        }
    }
}

```

FIG. 19C

TU

```

transform_unit( x0, y0, currTransformUnitSize ) {
    if( currTransformUnitSize > MinTransformUnitSize &&
        currTransformUnitSize <= MaxTransformUnitSize)
        split_transform_unit_flag
        if( split_transform_unit_flag ) {
            splitTransformUnitSize = currTransformUnitSize >> 1
            x1 = x0 + splitTransformUnitSize
            y1 = y0 + splitTransformUnitSize
            if( !entropy_coding_mode_flag )
                getVLCCBP( )
            transform_unit( x0, y0, split TransformUnitSize )
            if( x1 < PicWidthInSamplesL )
                transform_unit( x1, y0, split TransformUnitSize )
            if( y1 < PicHeightInSamplesL )
                transform_unit( x0, y1, split TransformUnitSize )
            if( x1 < PicWidthInSamplesL && y1 < PicHeightInSamplesL )
                transform_unit( x1, y1, split TransformUnitSize )
        } else {
            if( entropy_coding_mode_flag )
                coded_block_flag
                if( coded_block_flag ) {
                    rotational_transform_idx
                    if( !entropy_coding_mode_flag )
                        [residual_block_vlc( )]
                    else
                        residual_block_v2v( )
                }
            }
        }
}

```

d10

FIG. 20

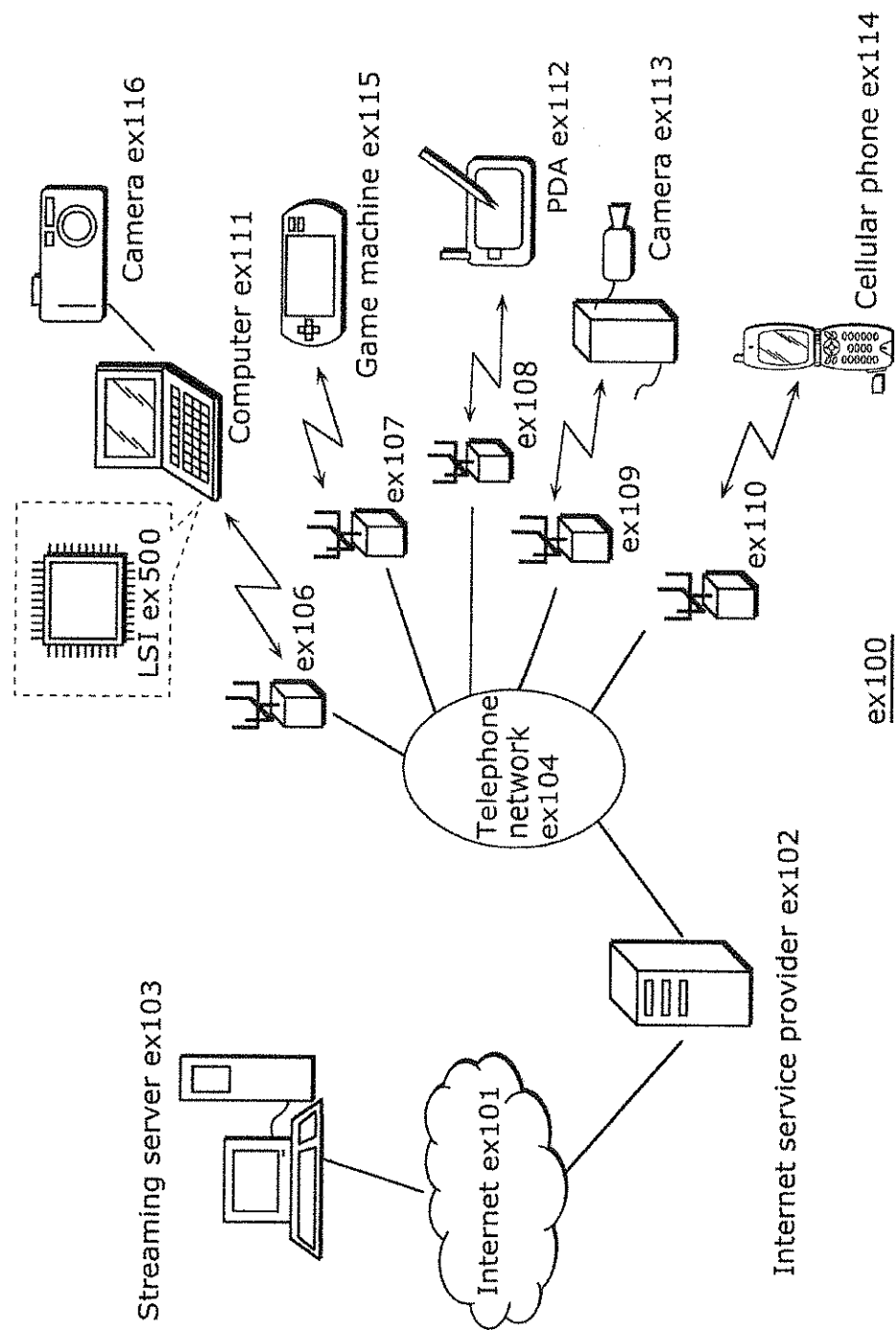


FIG. 21

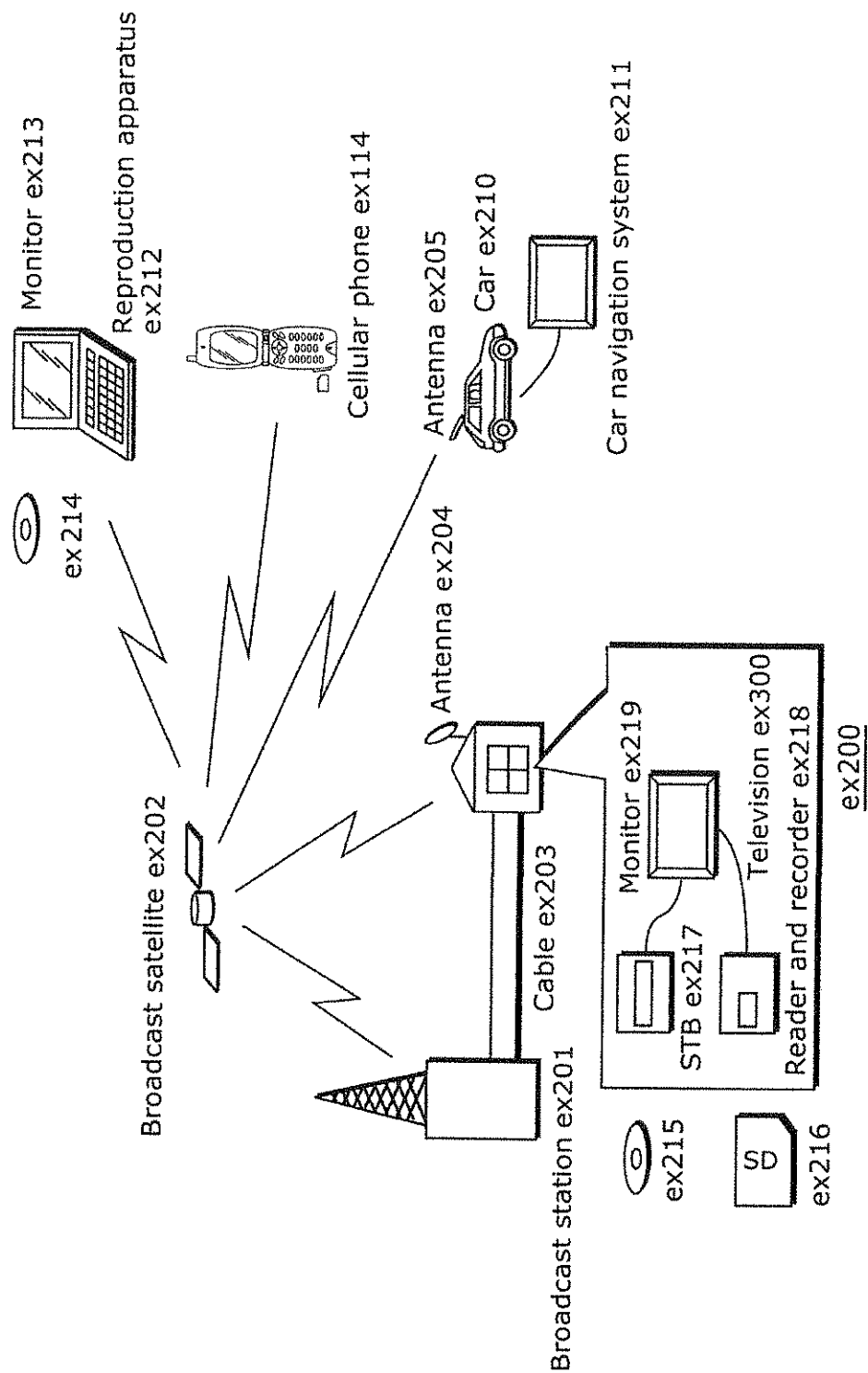


FIG. 22

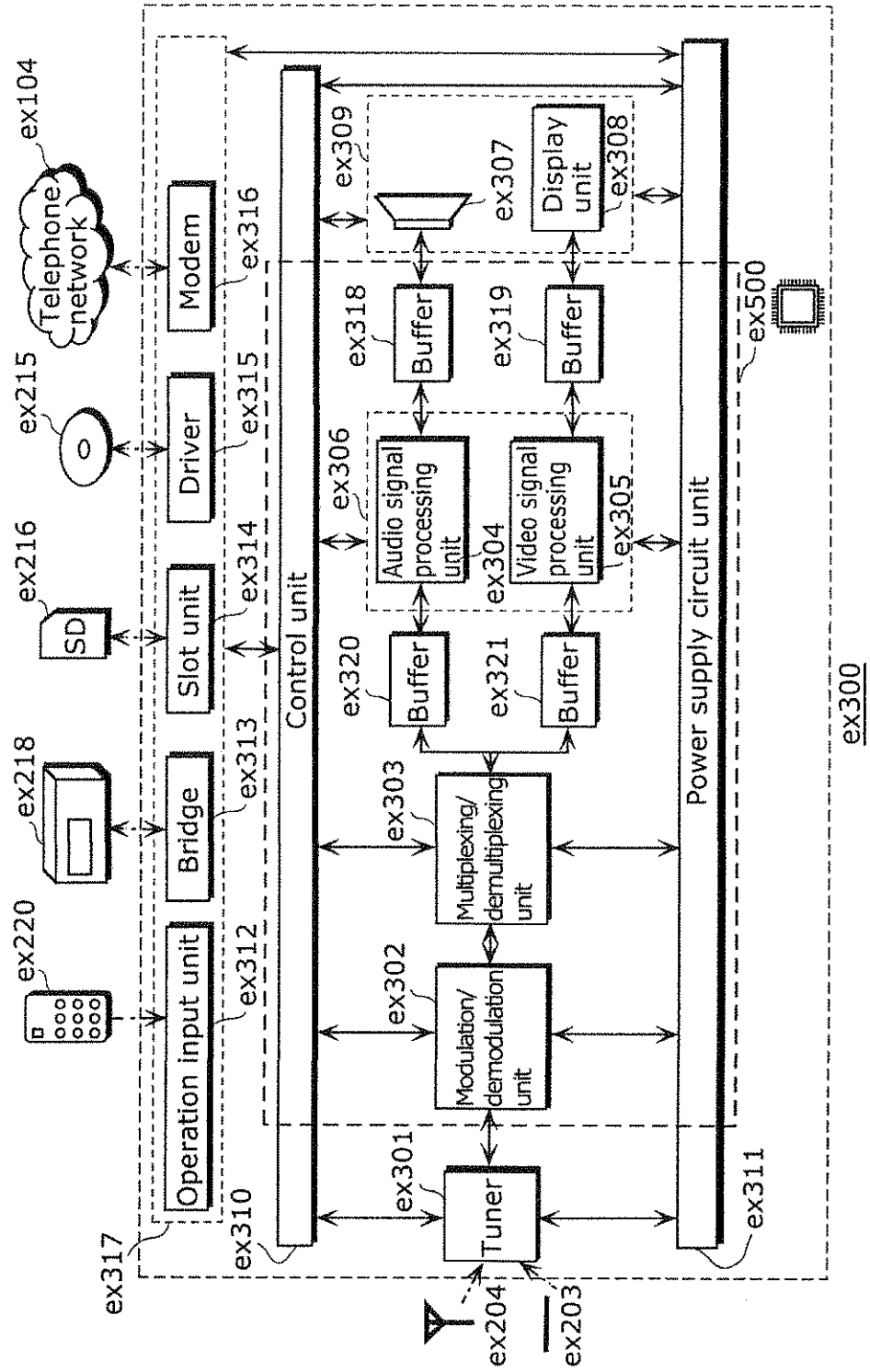


FIG. 23

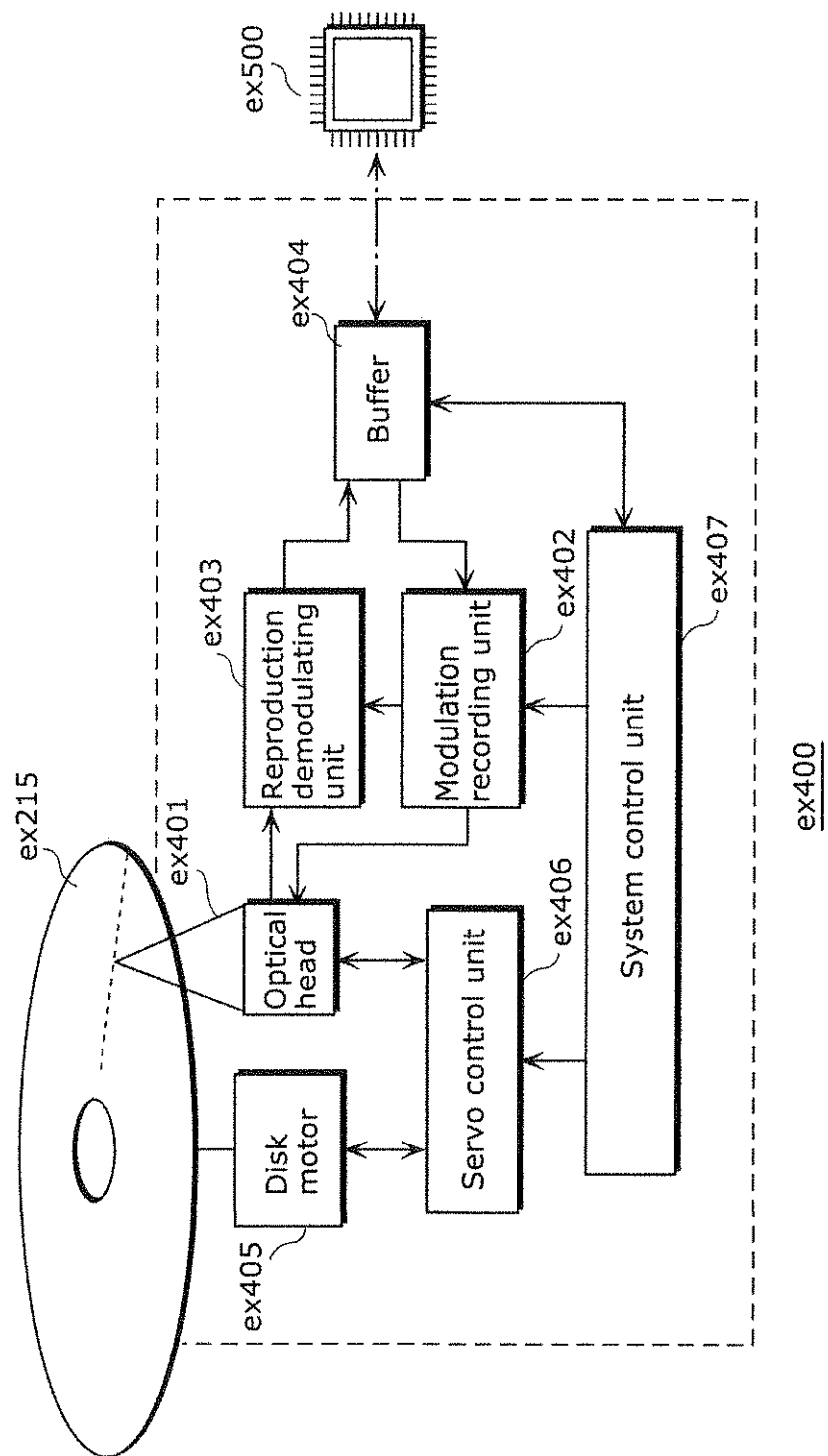


FIG. 24

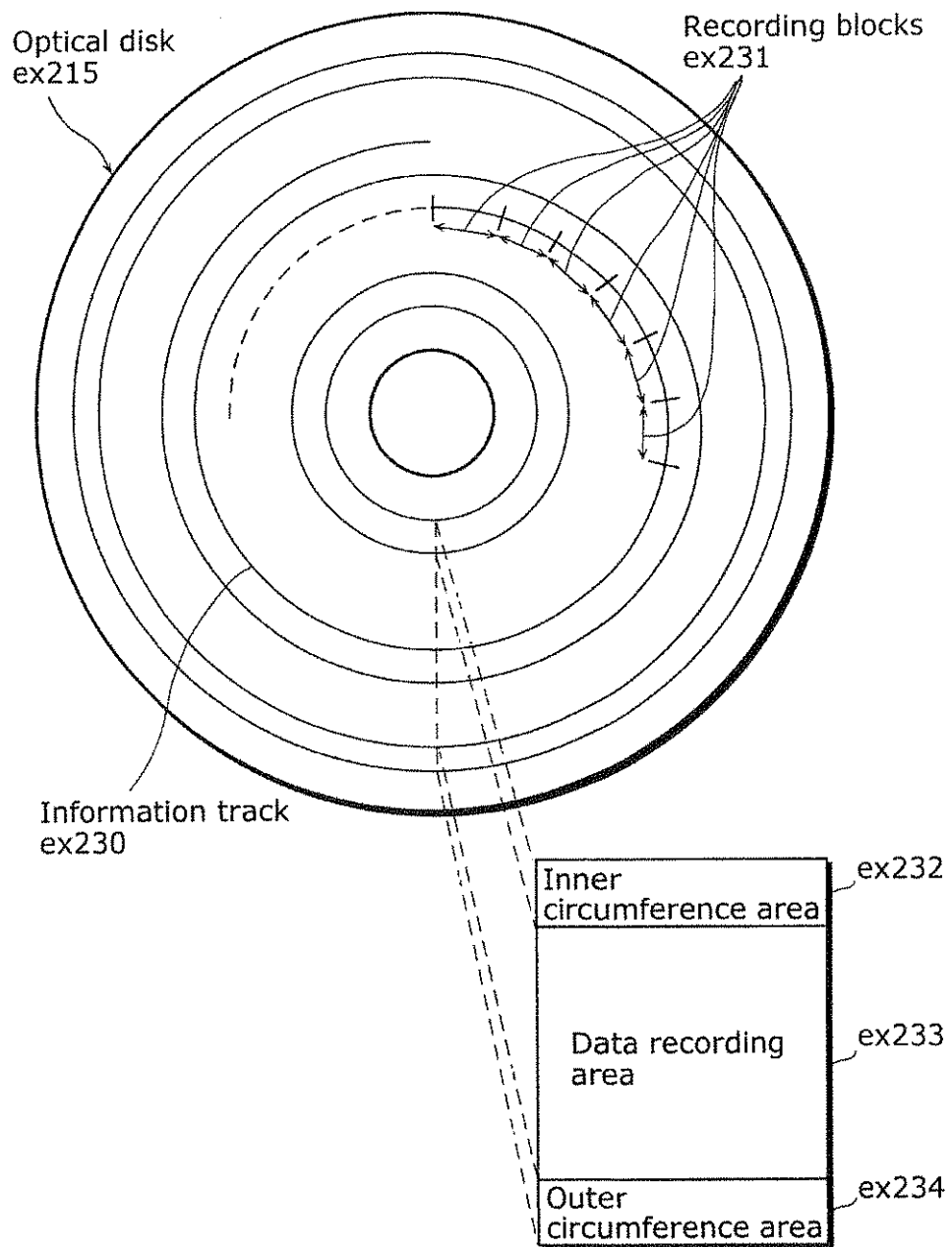


FIG. 25A

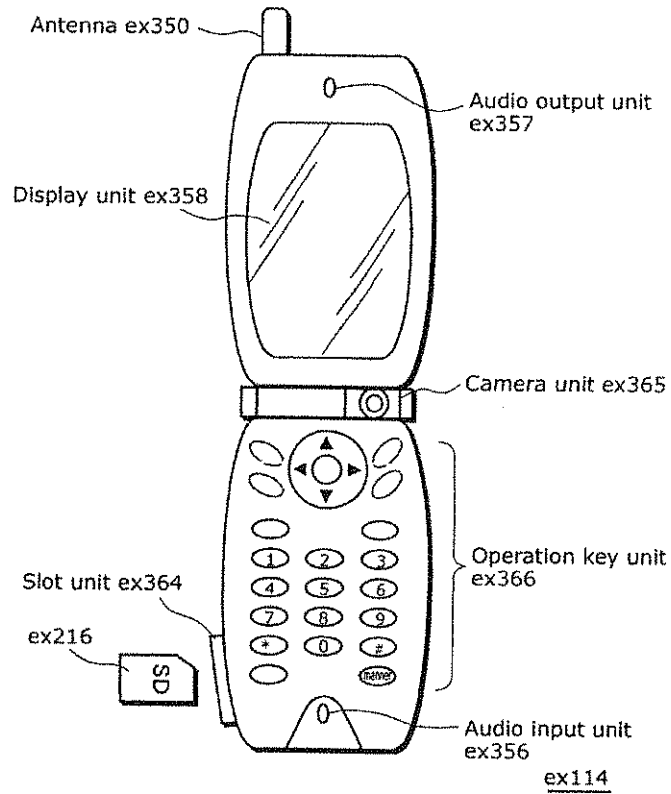


FIG. 25B

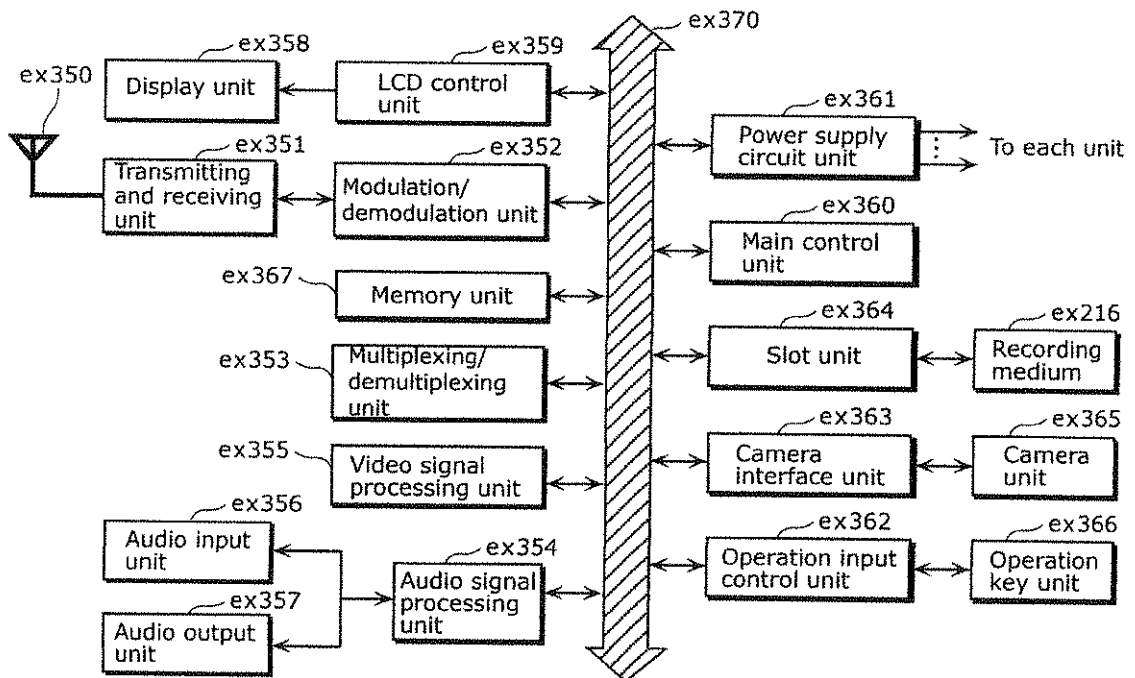


FIG. 26

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. 27

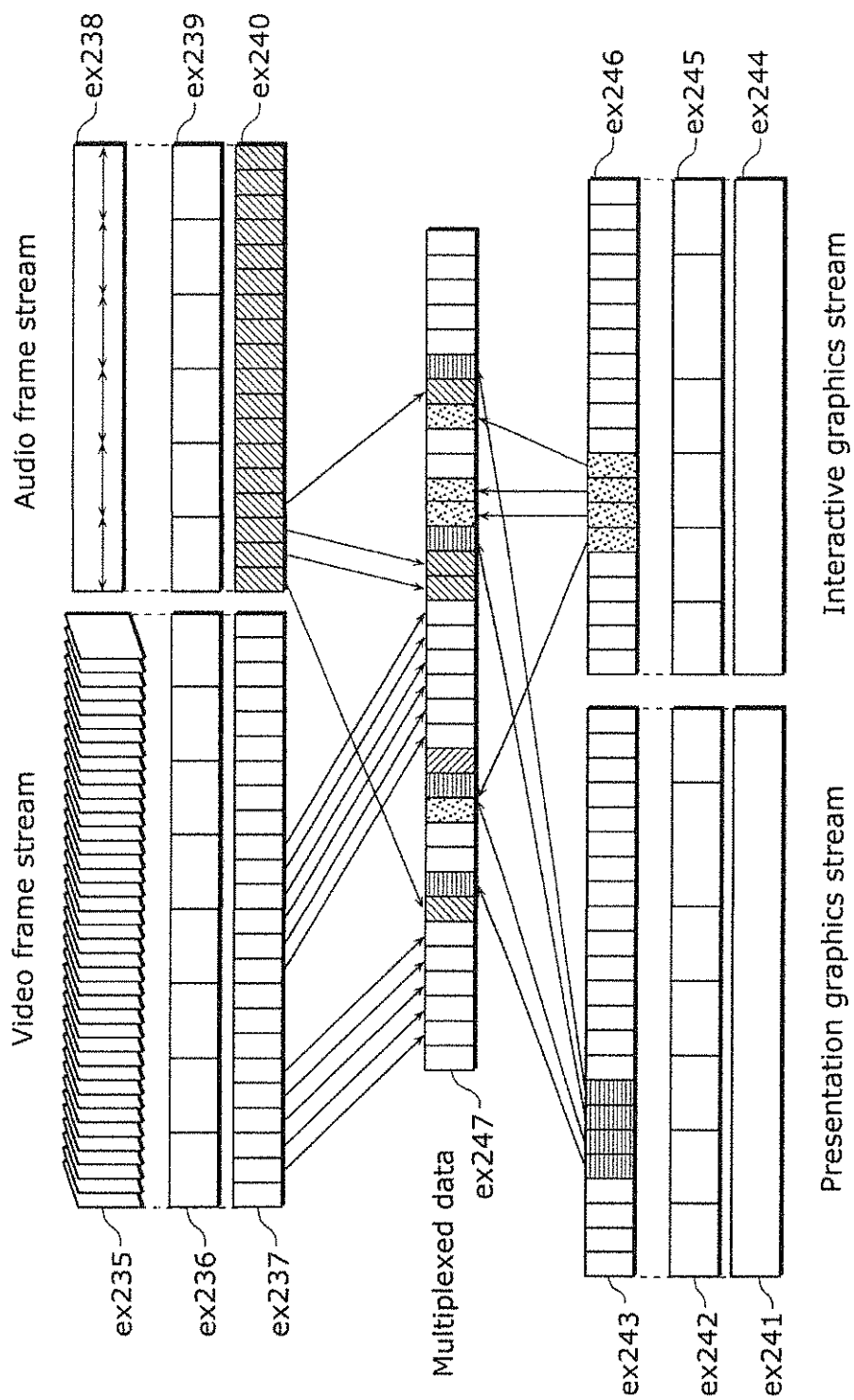


FIG. 28

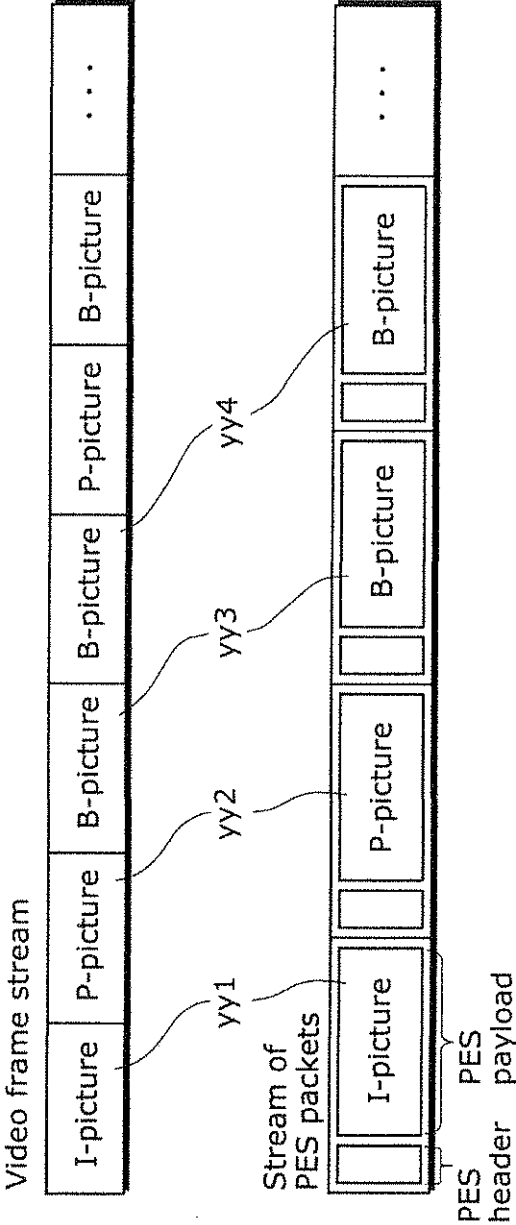


FIG. 29

Stream of TS packets

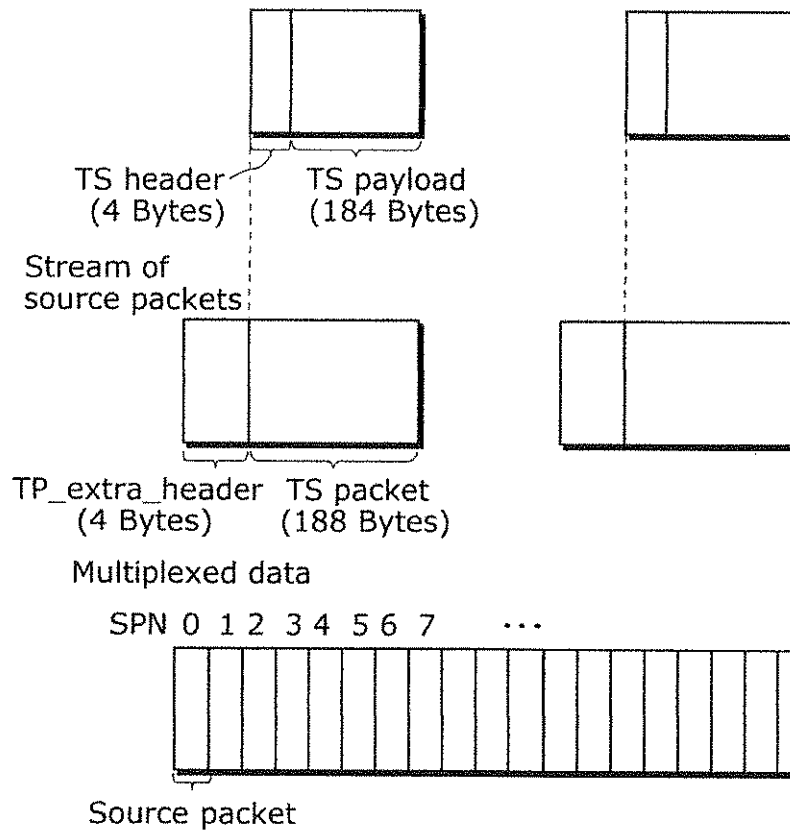


FIG. 30

Data structure of PMT

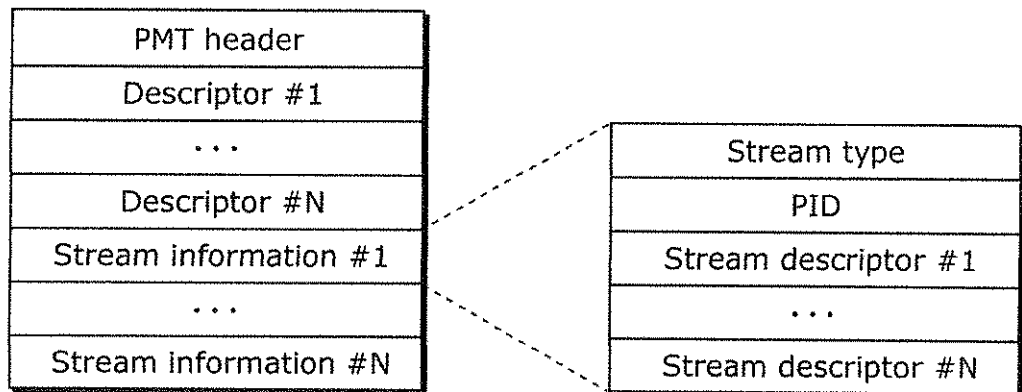


FIG. 31

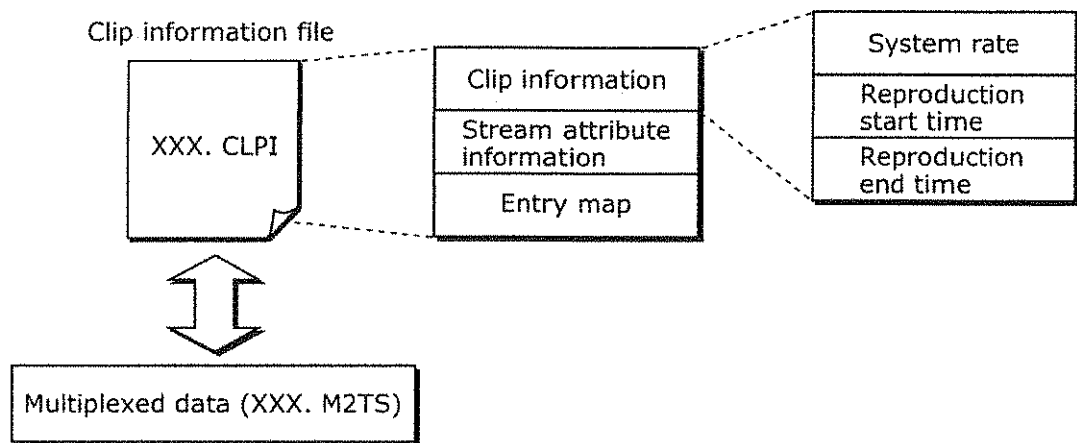


FIG. 32

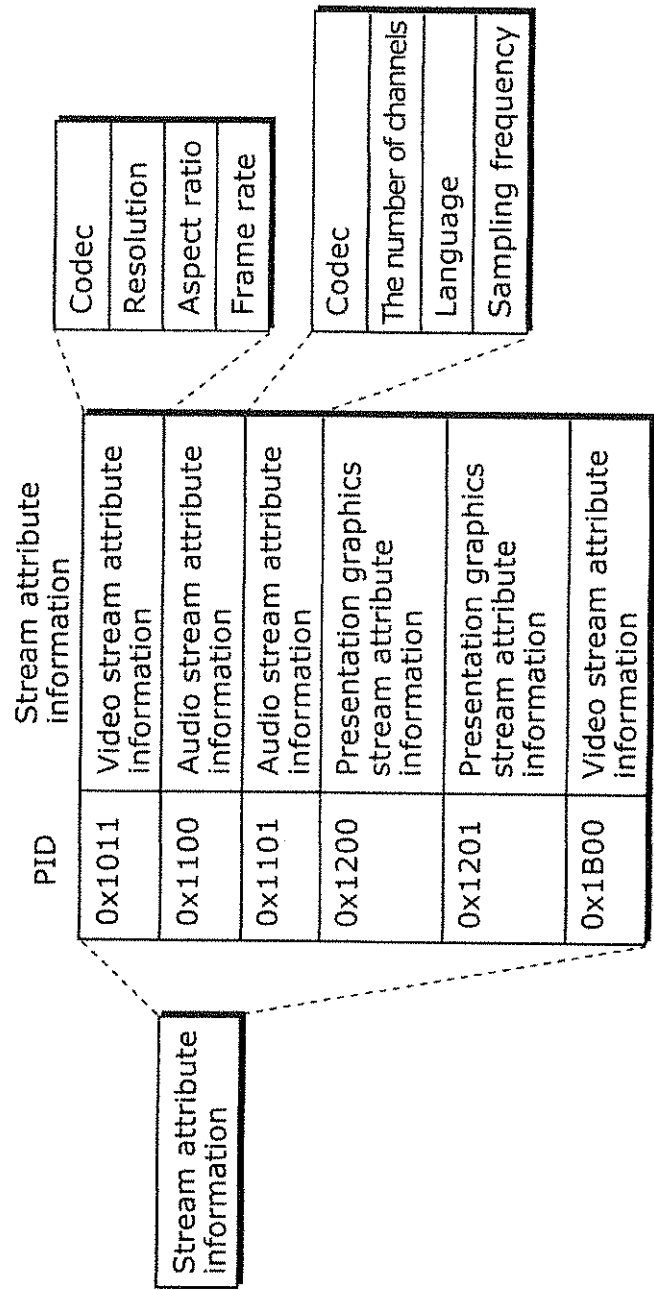


FIG. 33

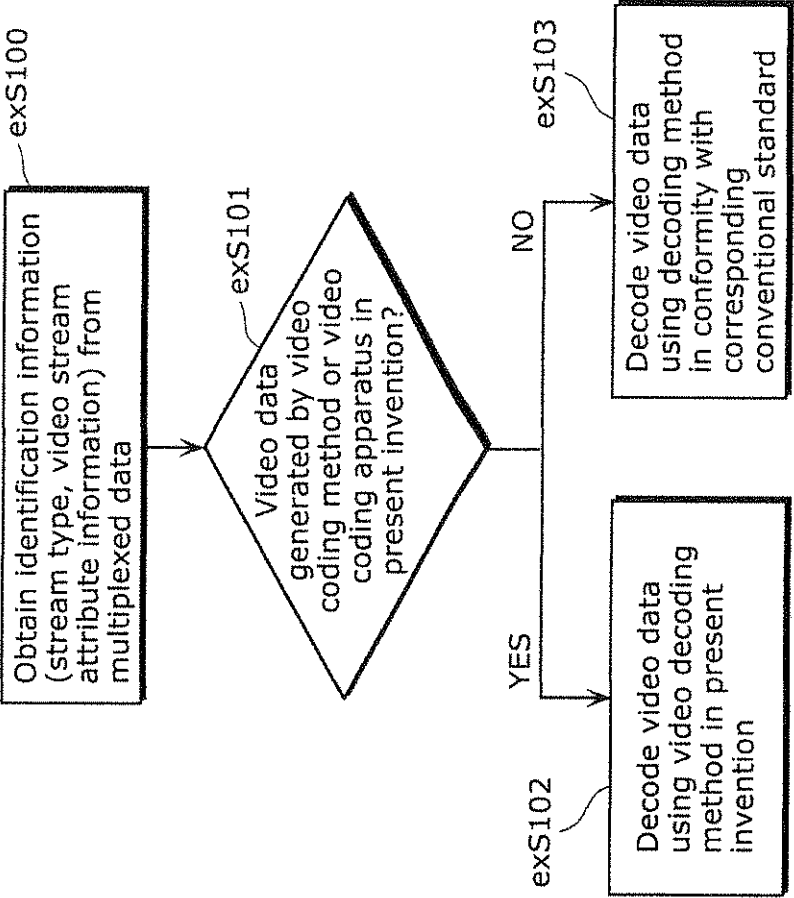


FIG. 34

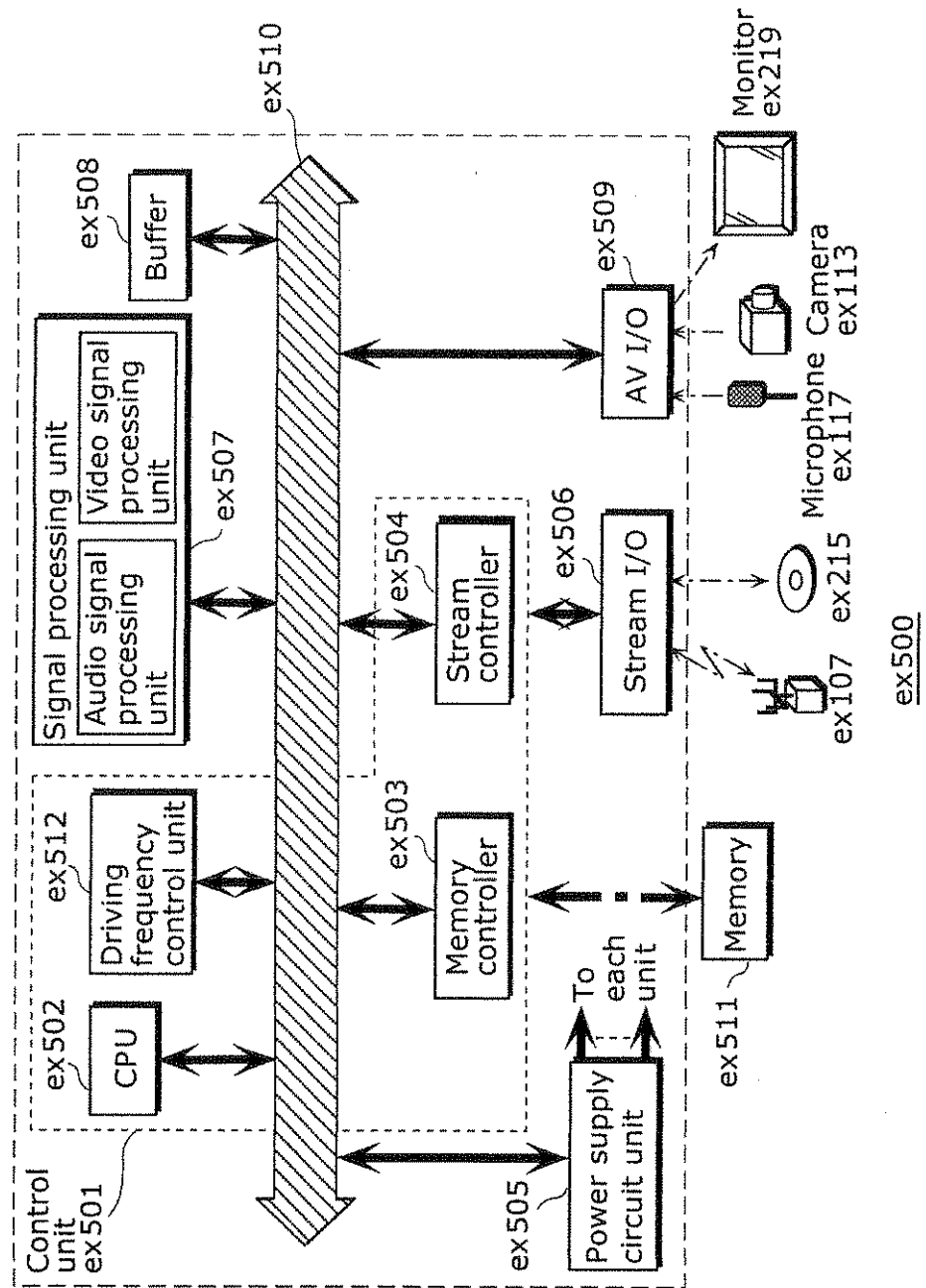


FIG. 35

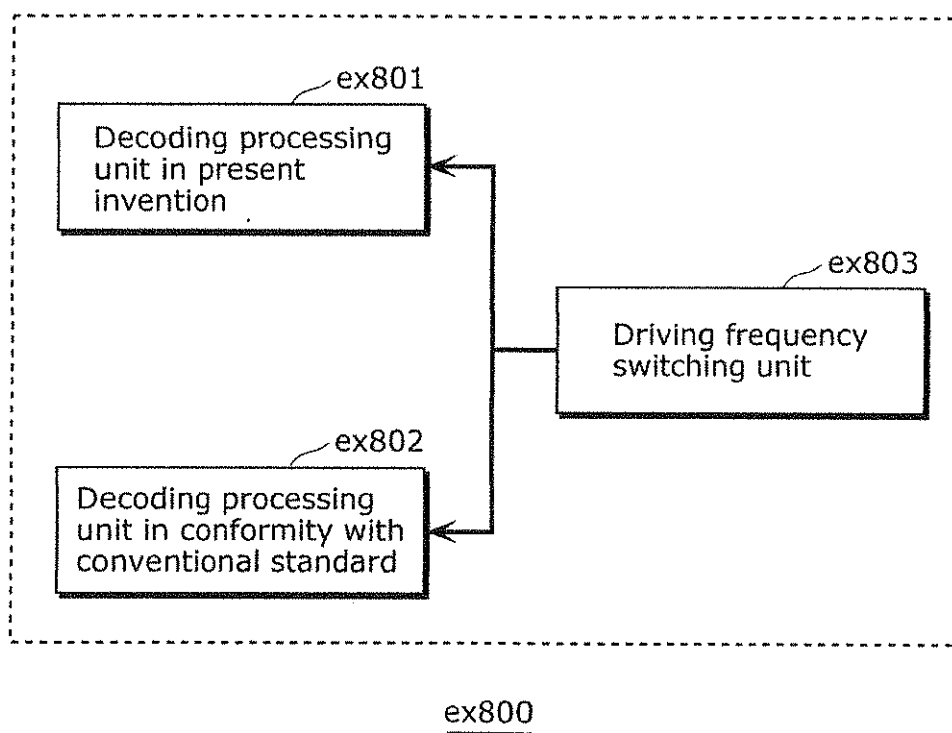


FIG. 36

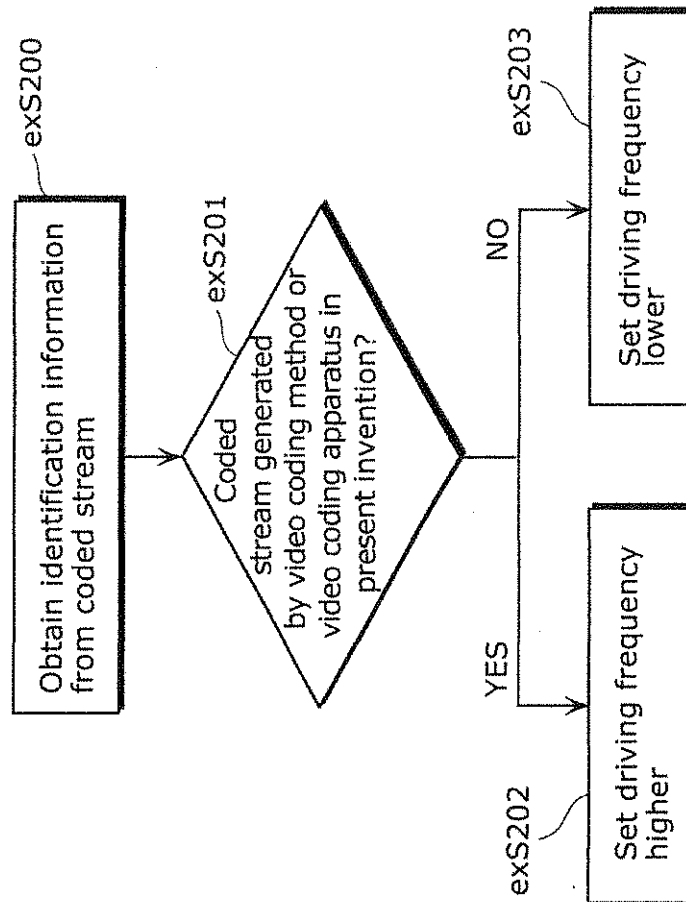


FIG. 37

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

FIG. 38A

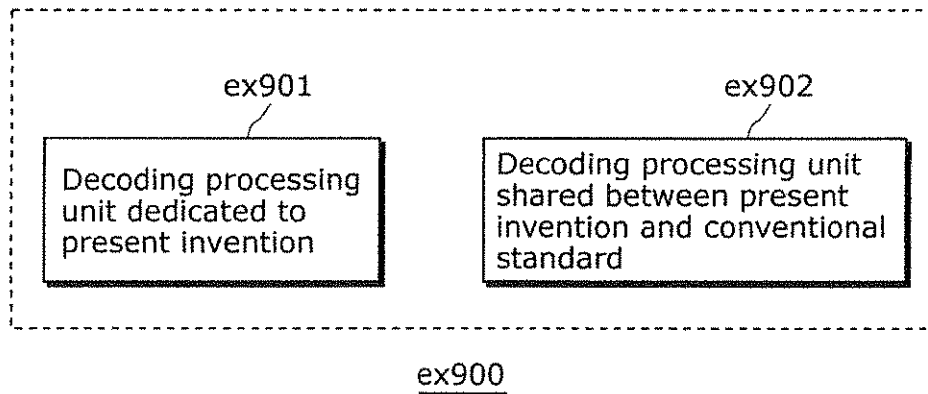


FIG. 38B

