

FIG. 1

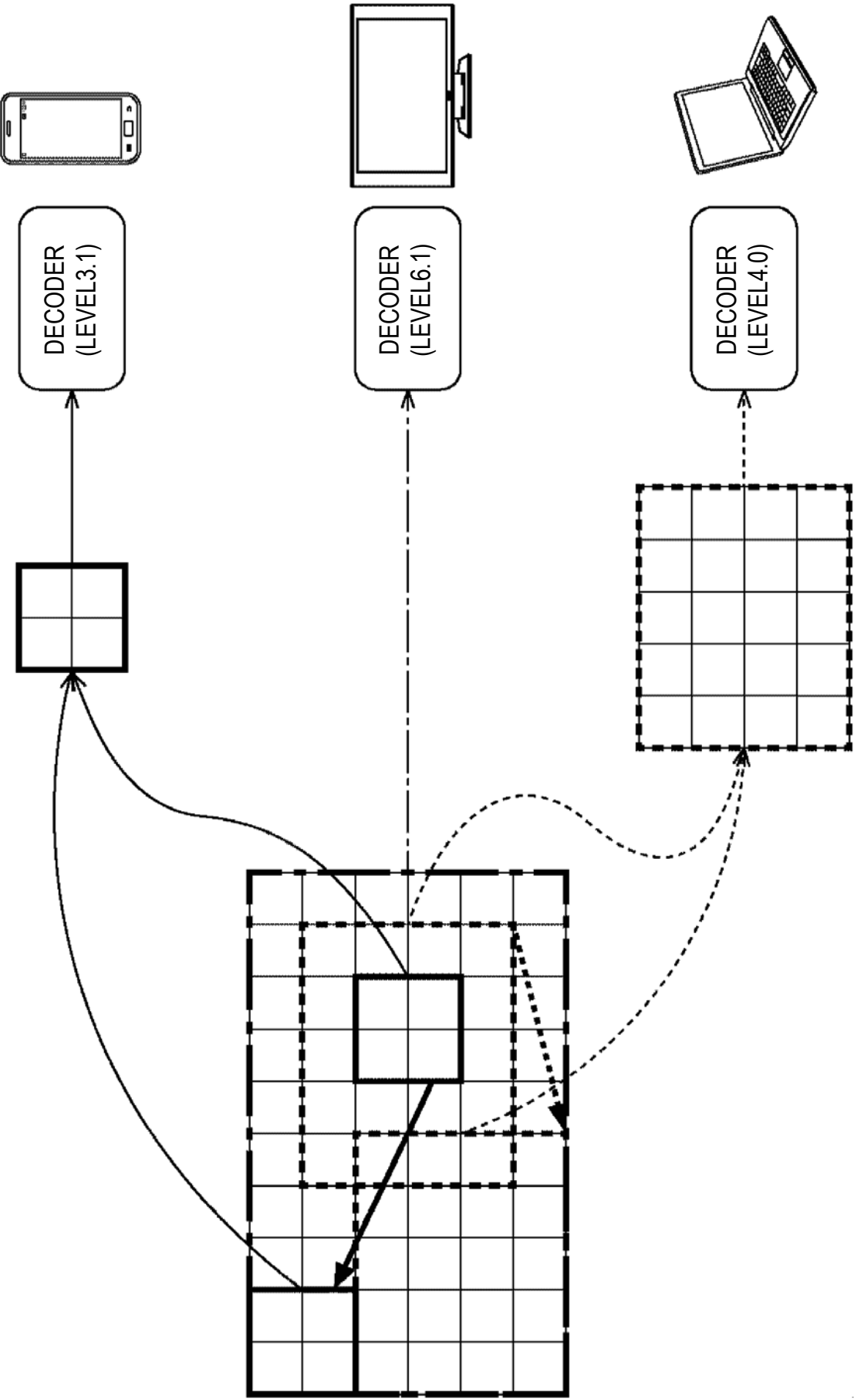


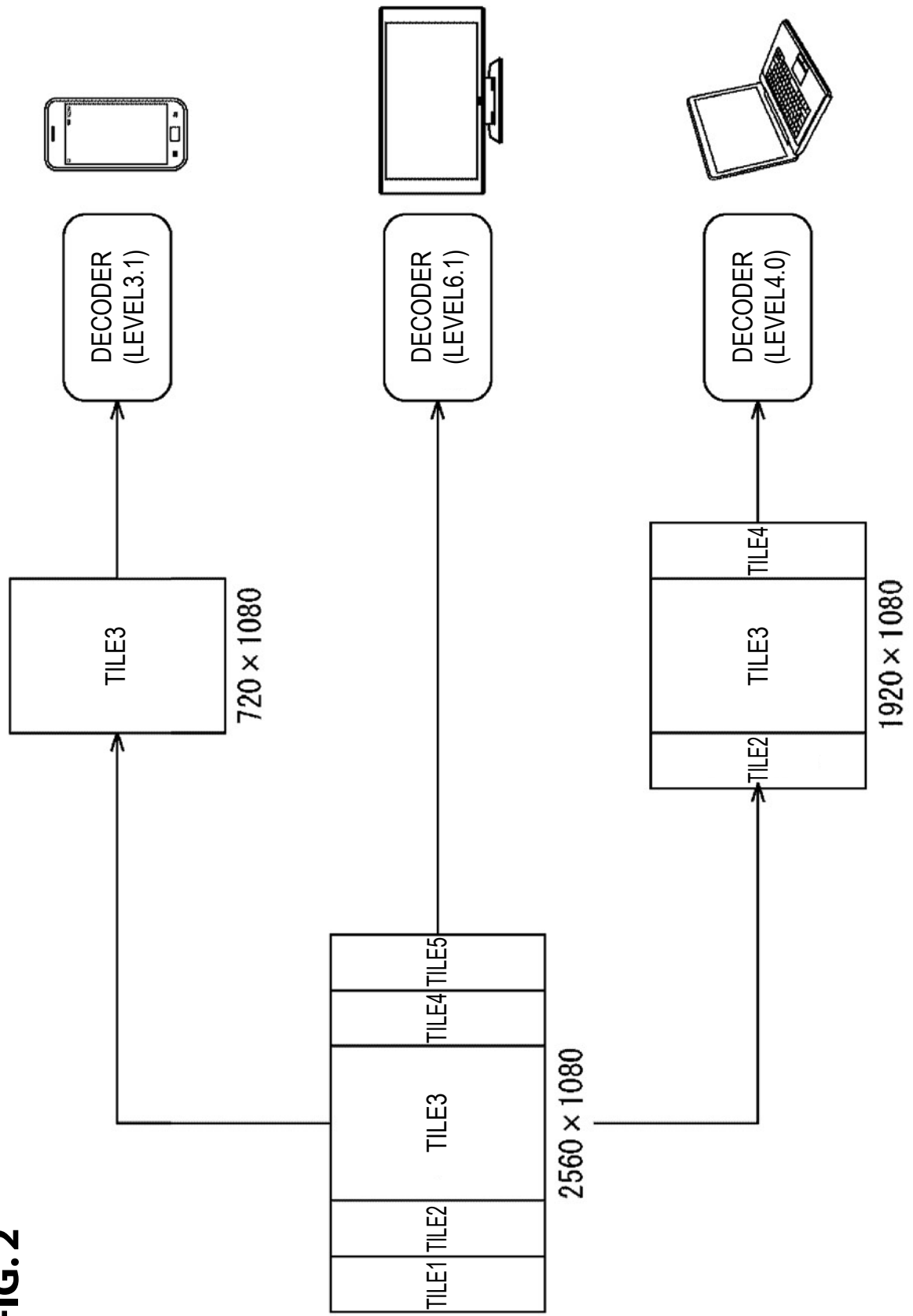
FIG. 2

FIG. 3

flags specifies the type of sub-sample information given in this box as follows:

0: NAL-unit-based sub-samples. A sub-sample contains one or more contiguous NAL units.

1: Decoding-unit-based sub-samples. A sub-sample contains exactly one decoding unit.

2: Tile-based sub-samples. A sub-sample either contains one tile and the associated non-VCL NAL units, if any, of the VCL NAL unit(s) containing the tile, or contains one or more non-VCL NAL units.

3: CTU-row-based sub-samples. A sub-sample either contains one CTU row within a slice and the associated non-VCL NAL units, if any, of the VCL NAL unit(s) containing the CTU row or contains one or more non-VCL NAL units. This type of sub-sample information shall not be used when `entropy_coding_sync_enabled_flag` is equal to 0.

4: Slice-based sub-samples. A sub-sample either contains one slice (where each slice may contain one or more slice segments, each of which is a NAL unit) and the associated non-VCL NAL units, if any, or contains one or more non-VCL NAL units.

FIG. 4

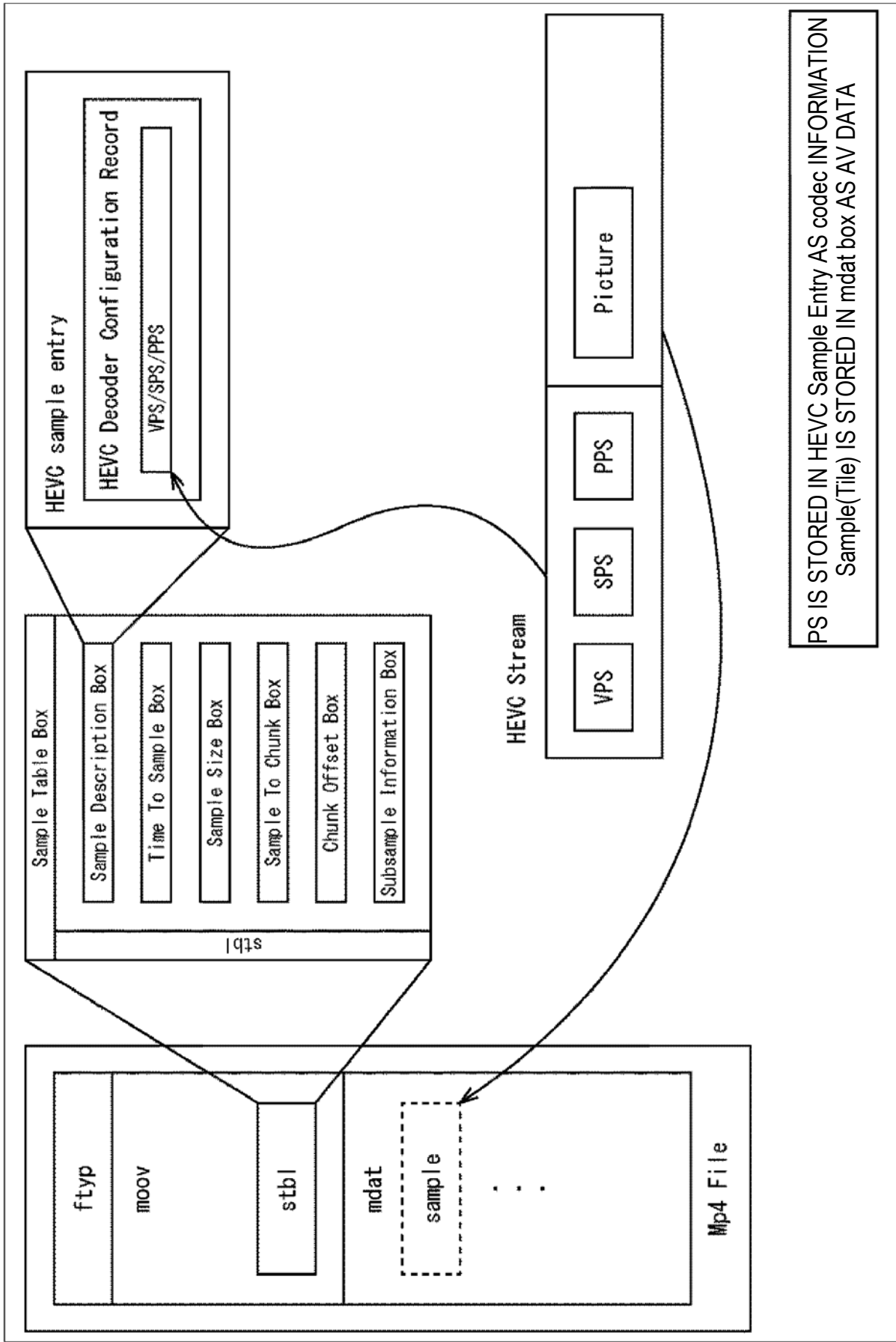


FIG. 5

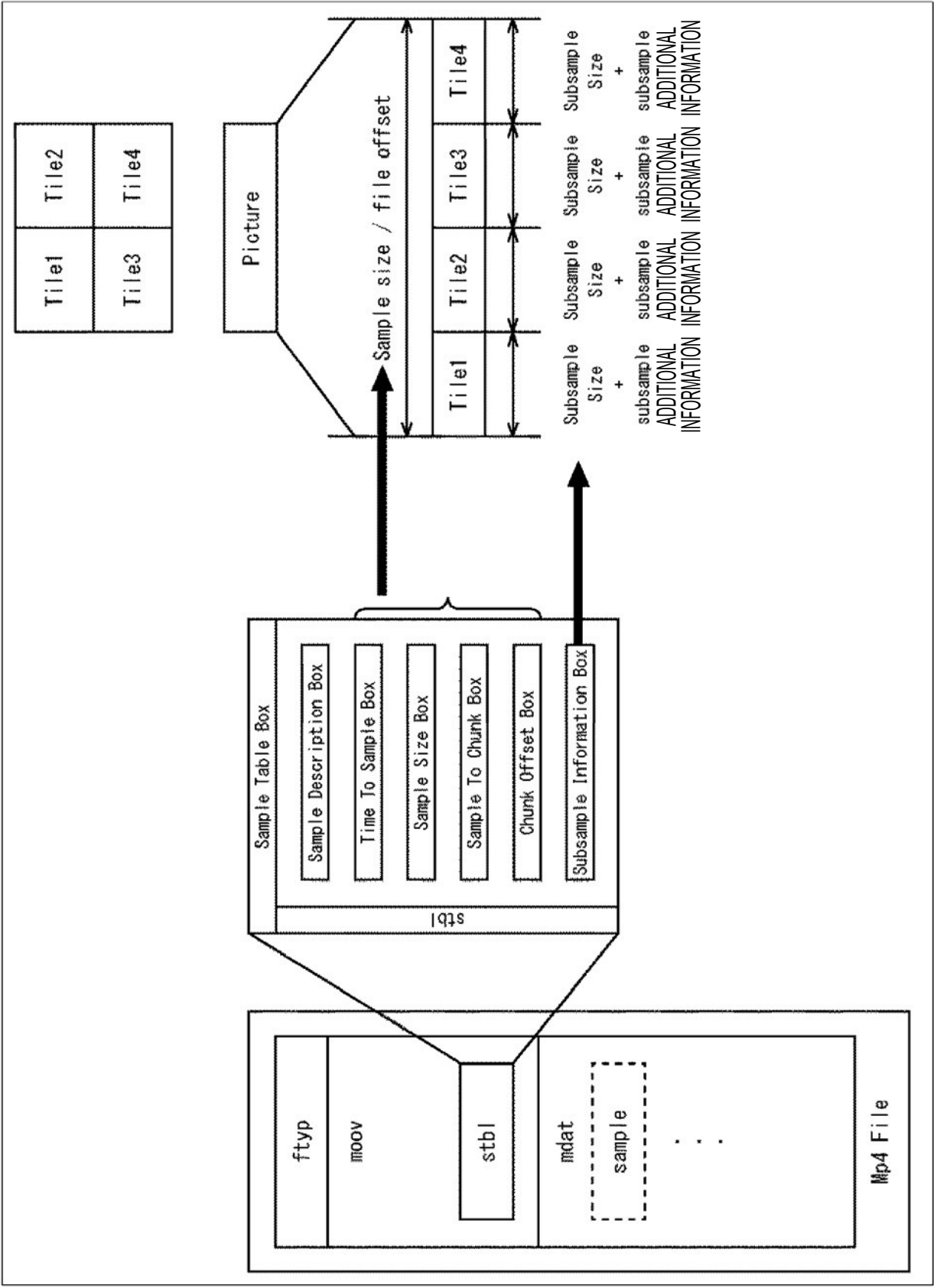


FIG. 6

ACCESS INFORMATION TO Tile
ACCESS UNIT OF MP4 IS sample
Subsample Information Box IS USED TO ACCESS Tile OR THE LIKE
STORING METHOD OF size OF Tile/LOCATION INFORMATION OF Tile IS DEFINED

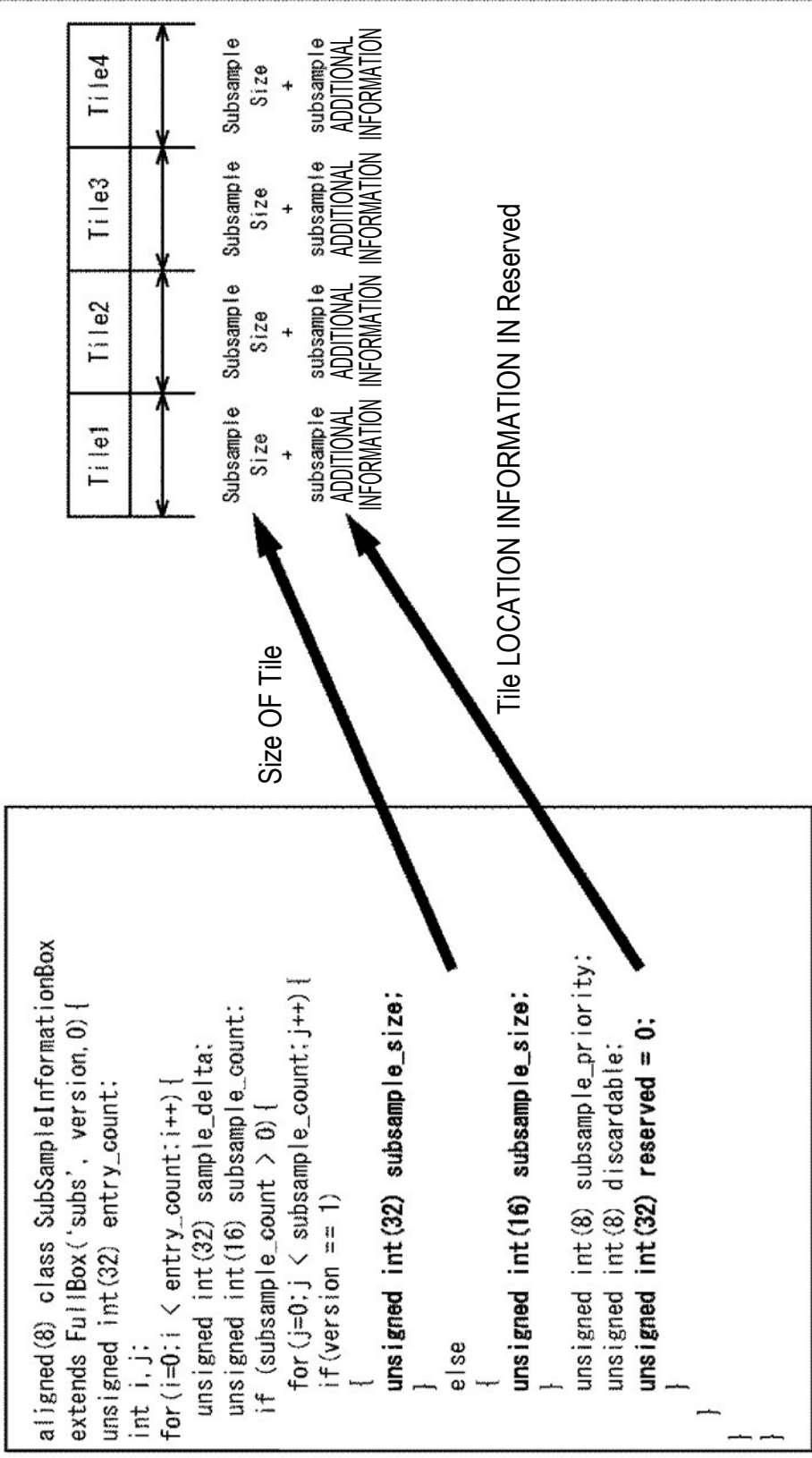


FIG. 7

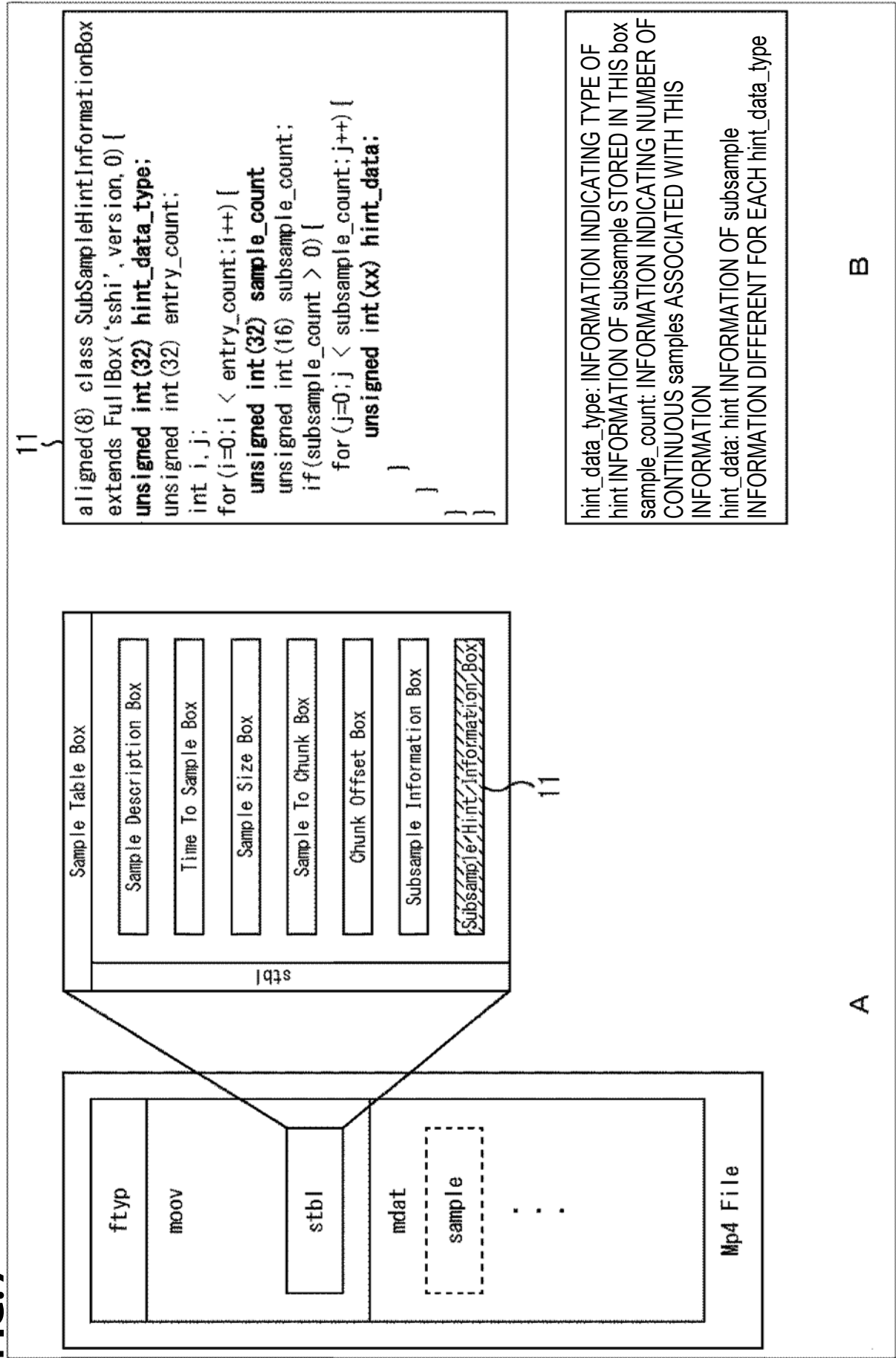


FIG. 8

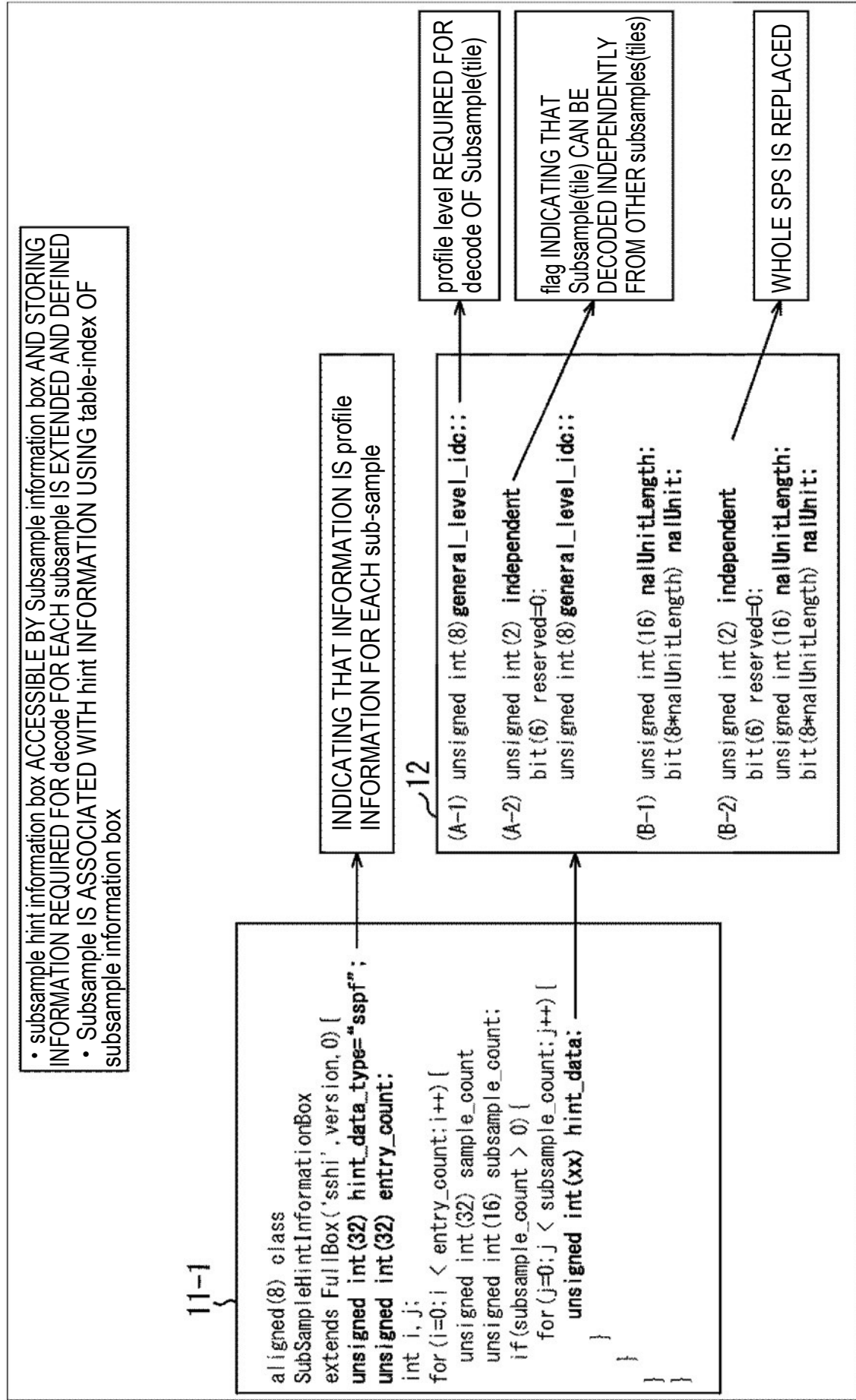


FIG. 9

Bit(2) semantics OF Independent

independent specifies that this tile has decoding dependencies only to the same tile in other samples, as follows:

0: the coding dependencies between this tile and other tiles in the same frame or previous frames is unknown,

1: there are no spatial coding dependencies between this tile and other tiles in the same frame, no temporal dependencies between this tile and the other tiles with same region in any reference frames but there can be coding dependencies between this tile and the tile with the same region in the reference frames

2: there are no coding dependencies between this tile and other tiles with same region in the same frame or in previous frames,

3: reserved.

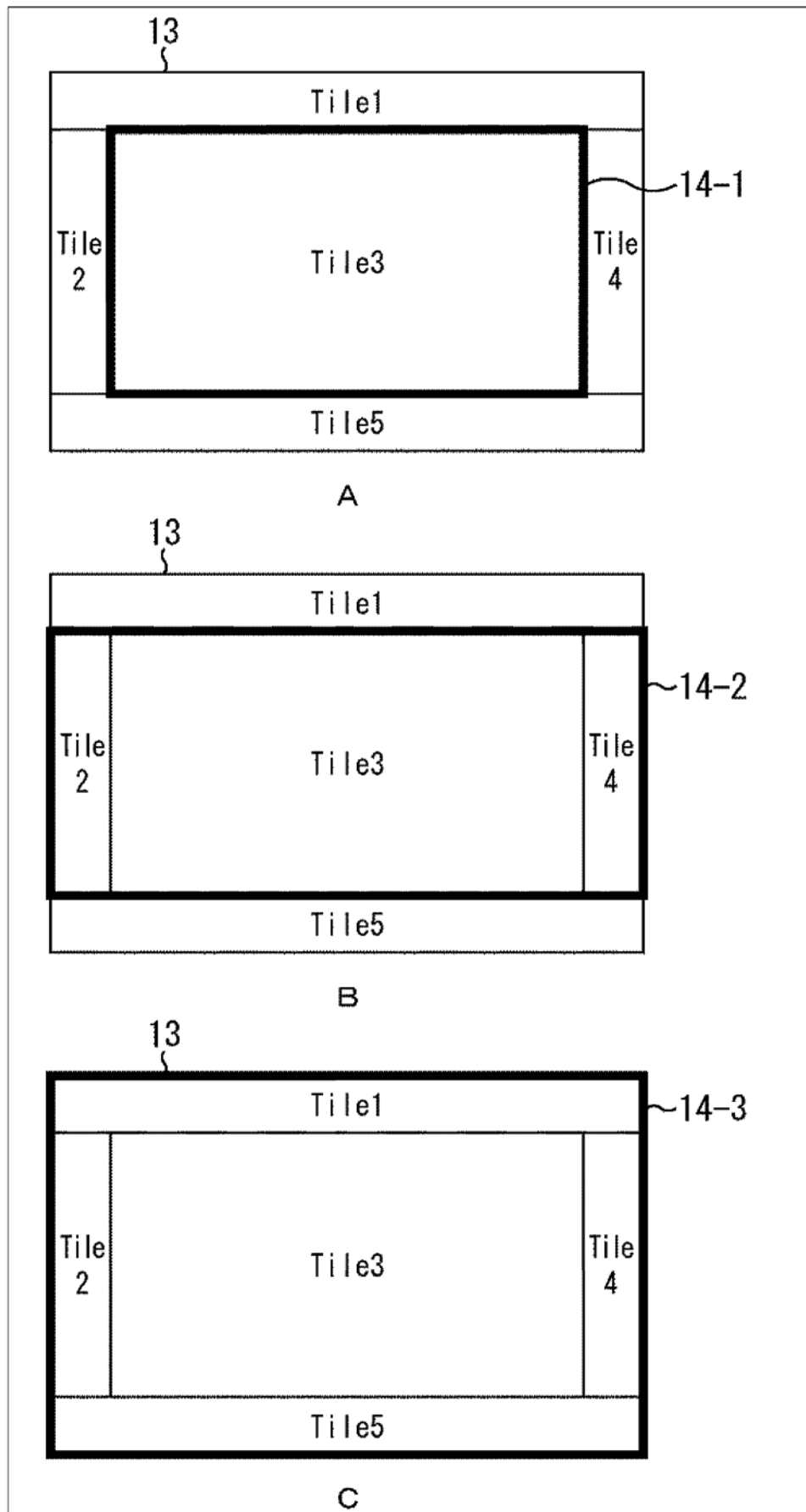
FIG. 10

FIG. 11

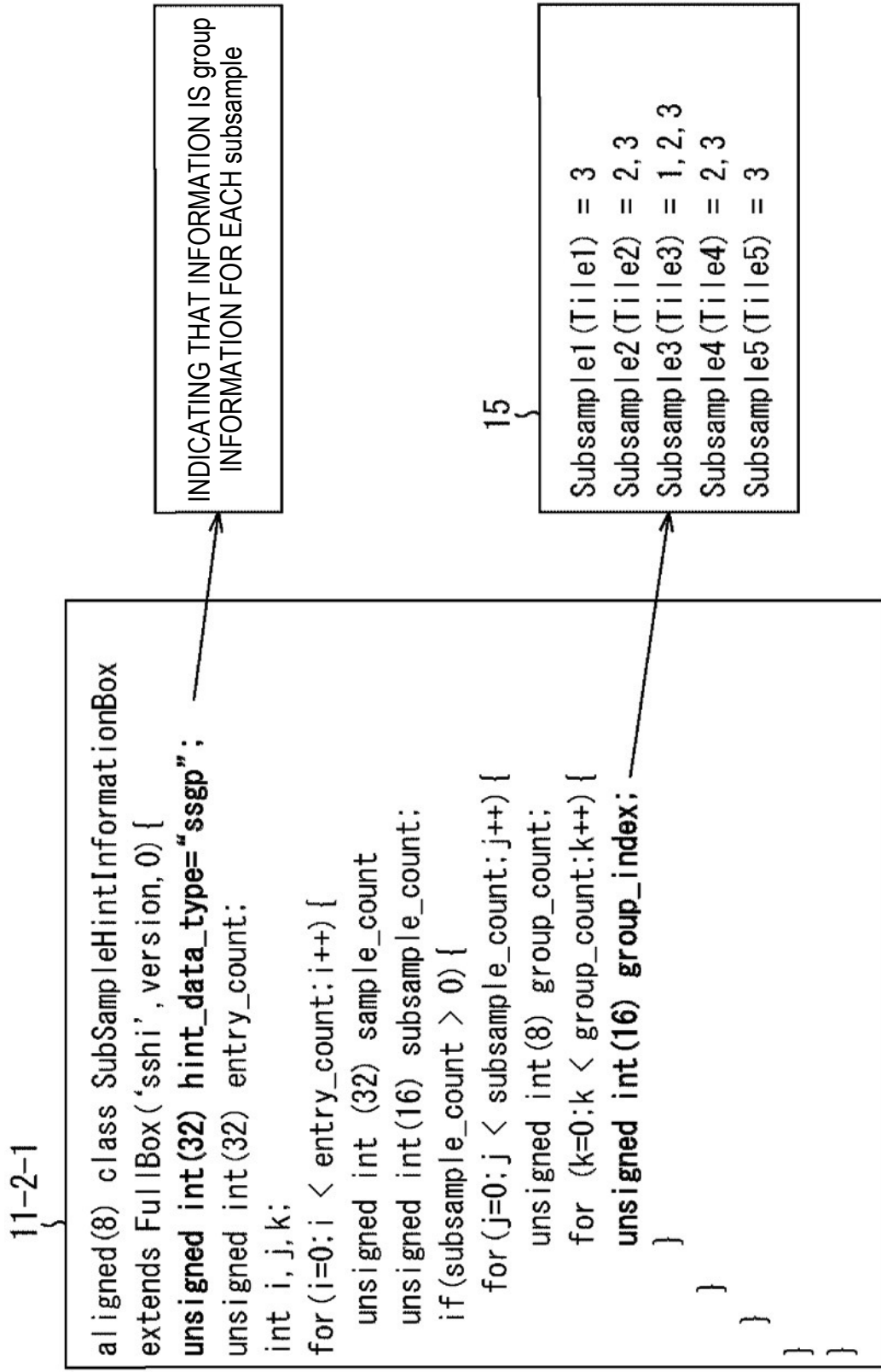


FIG. 12

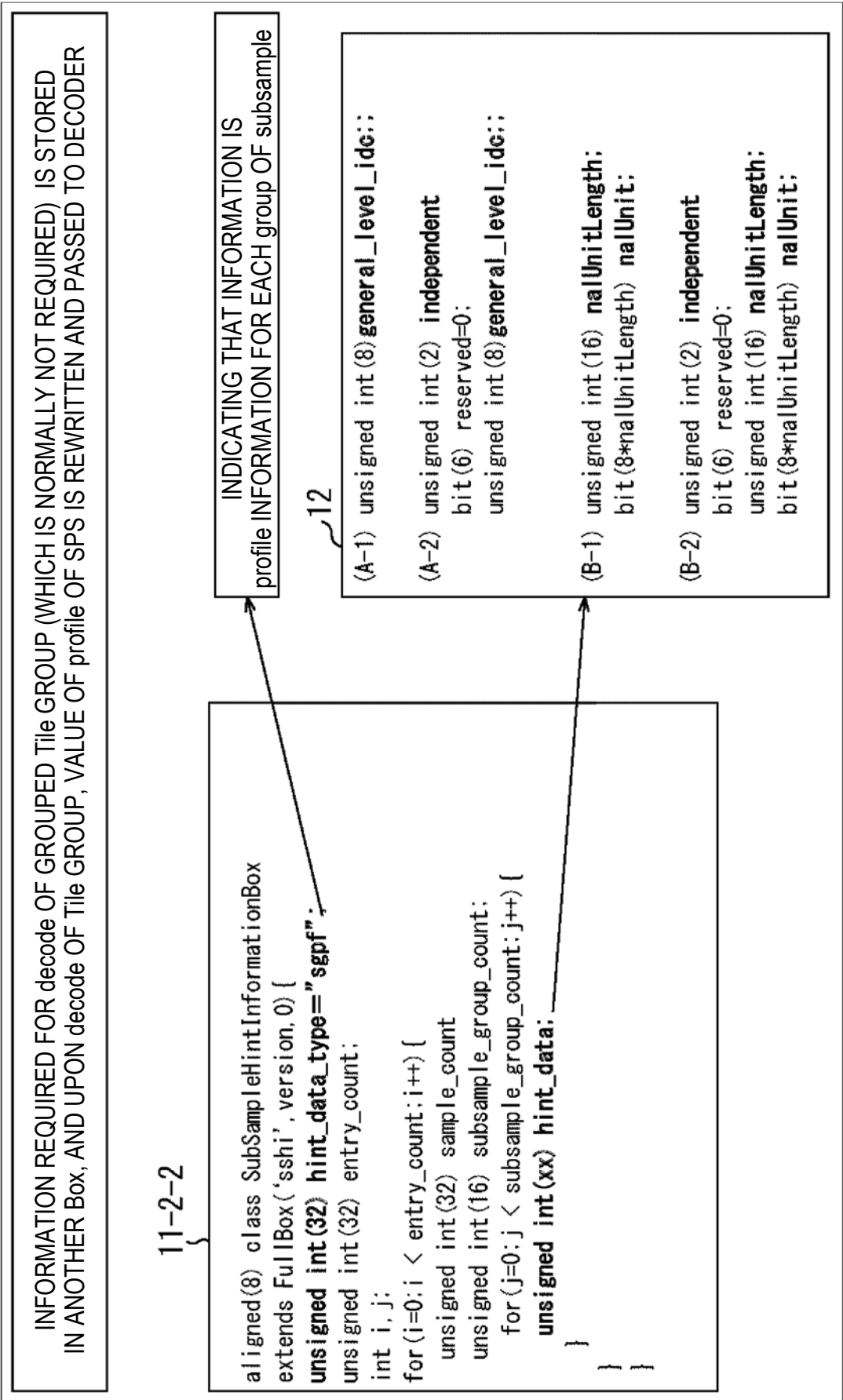


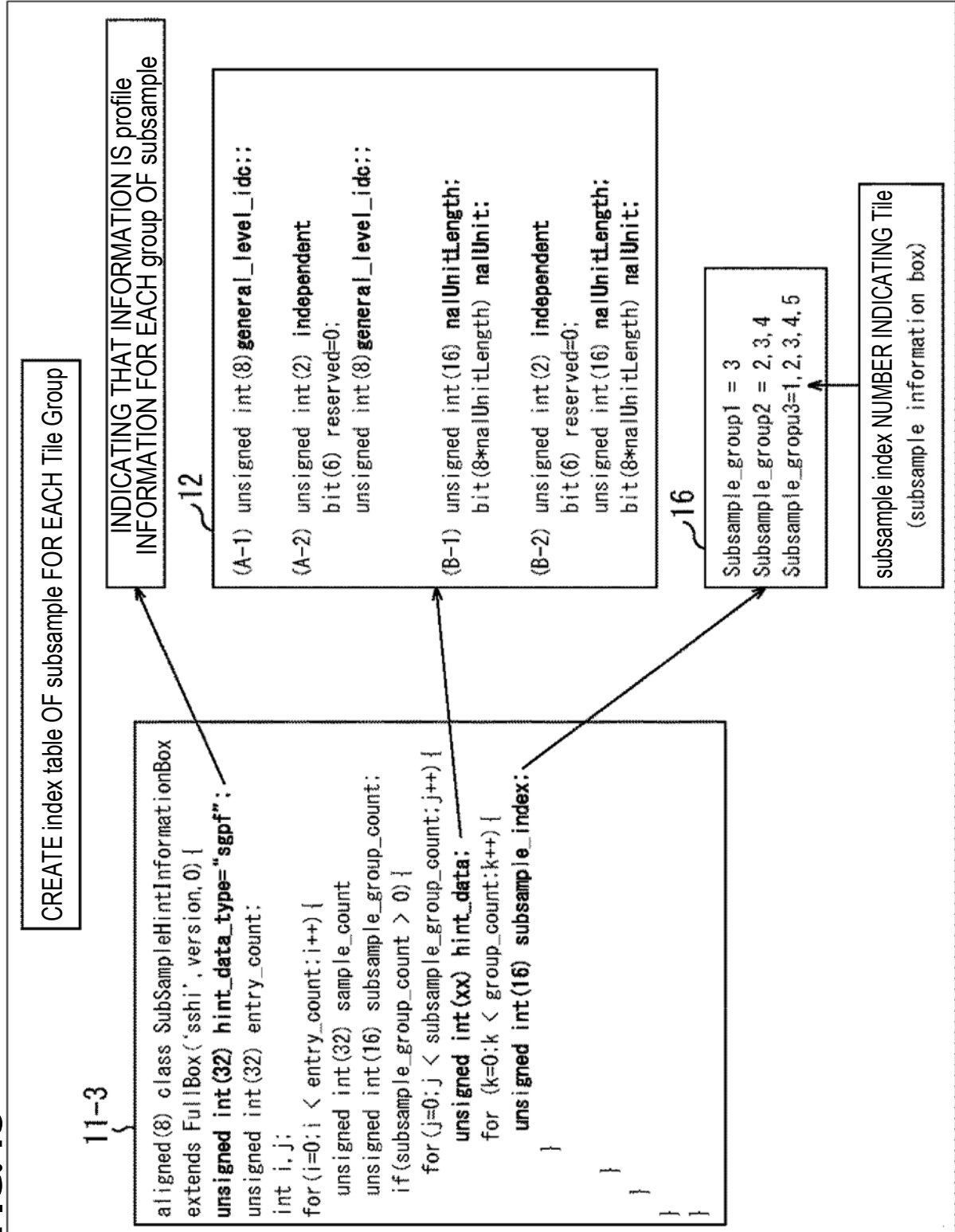
FIG. 13

FIG. 14

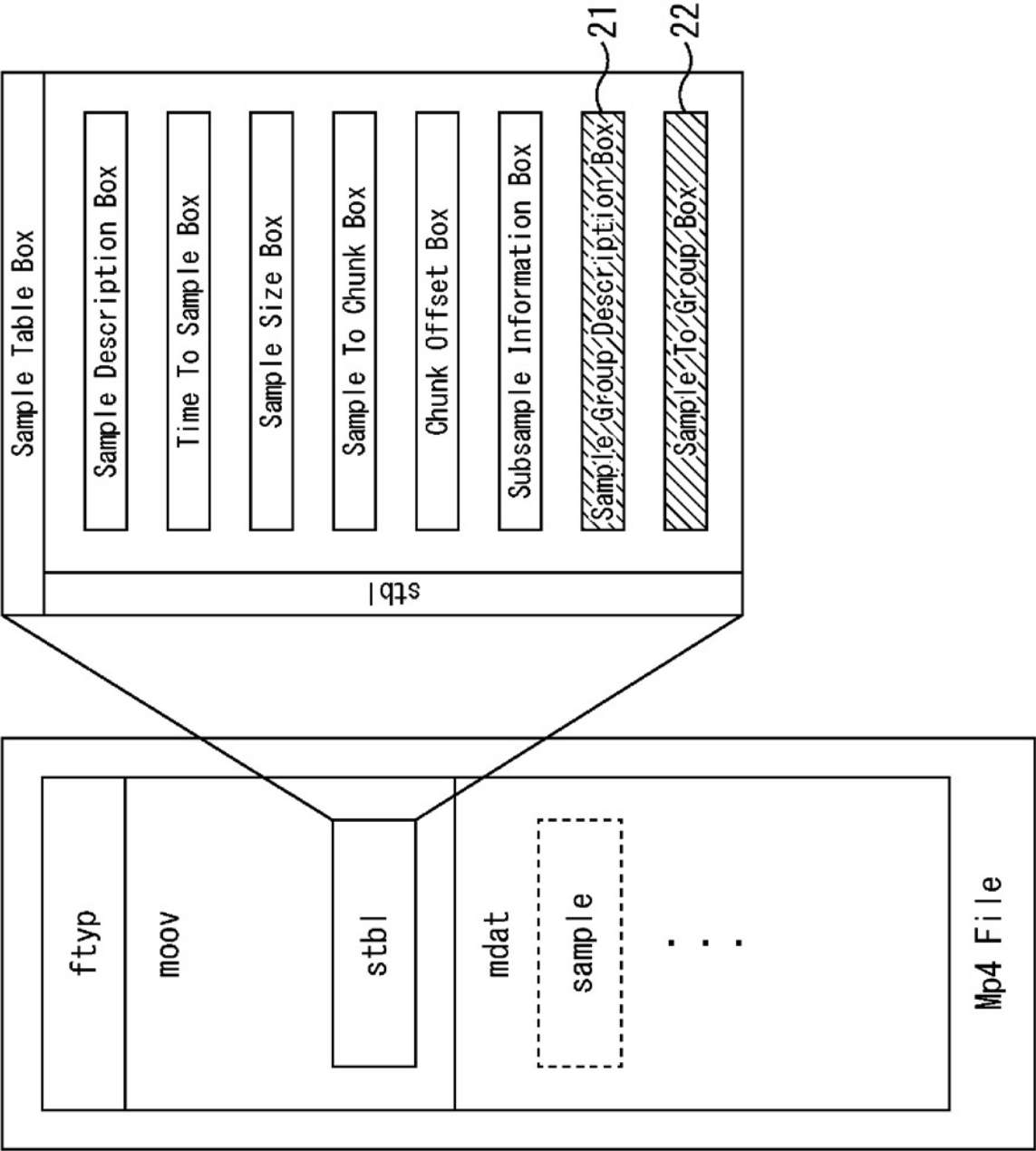


FIG. 15

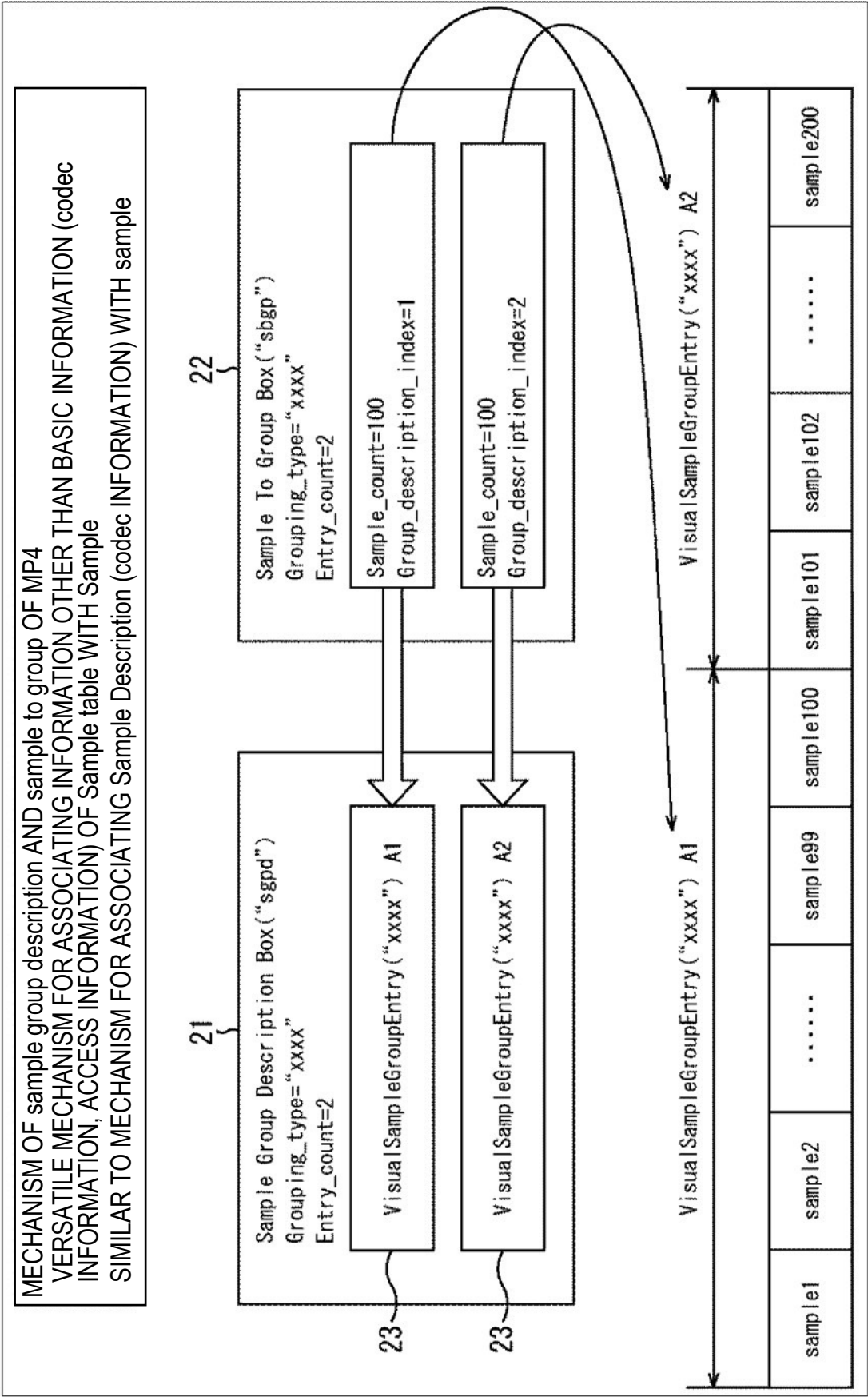


FIG. 16

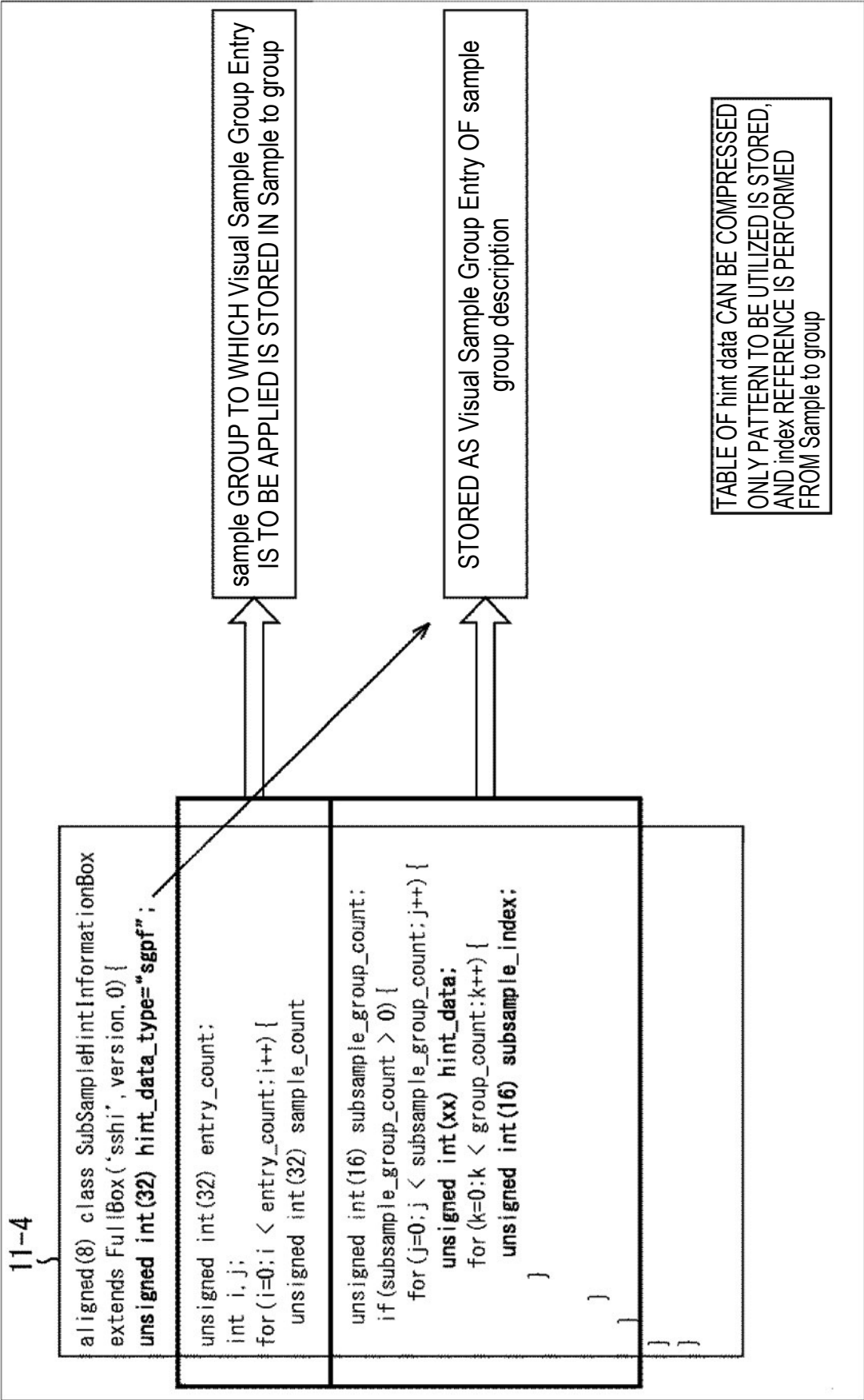
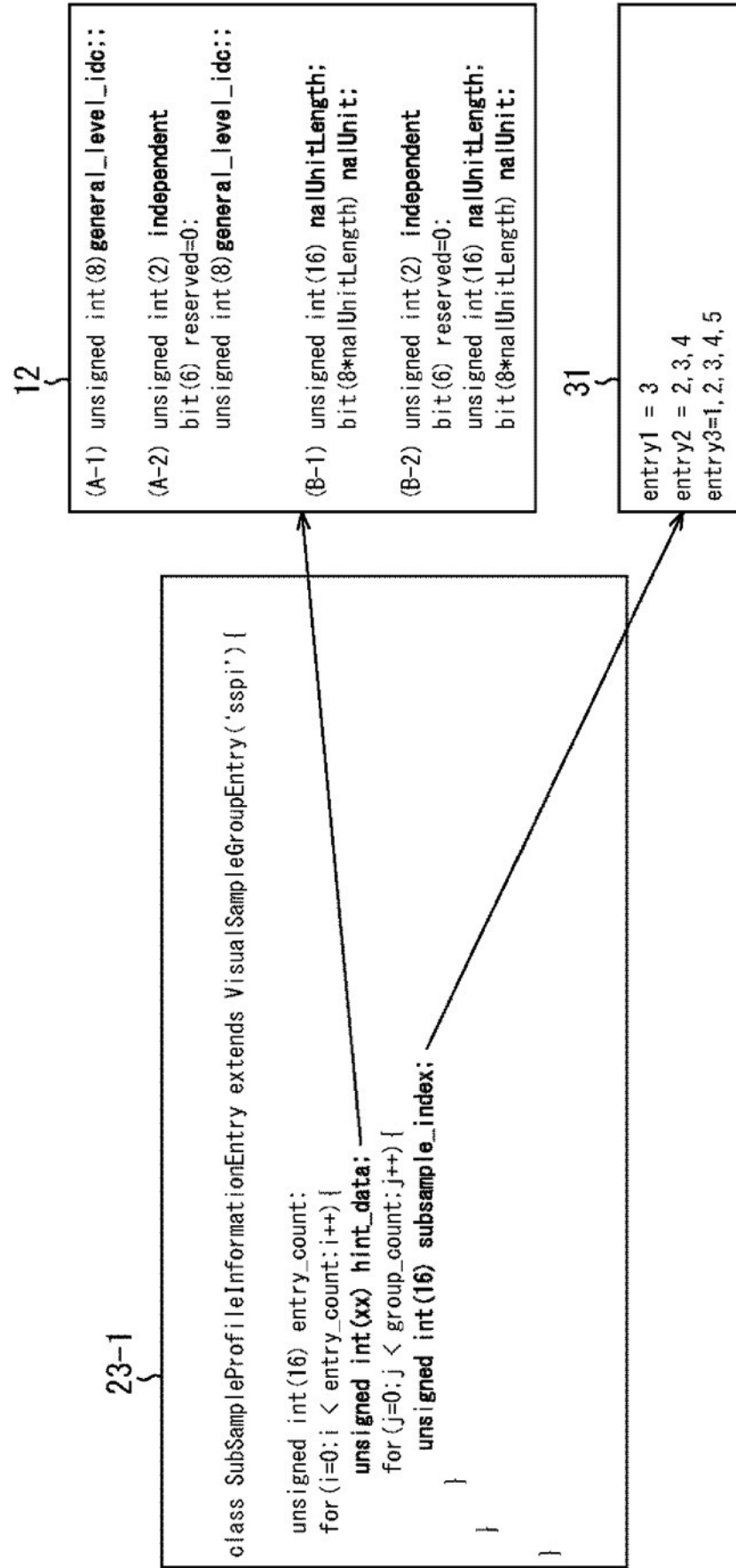


FIG. 17



18/49
FIG. 18

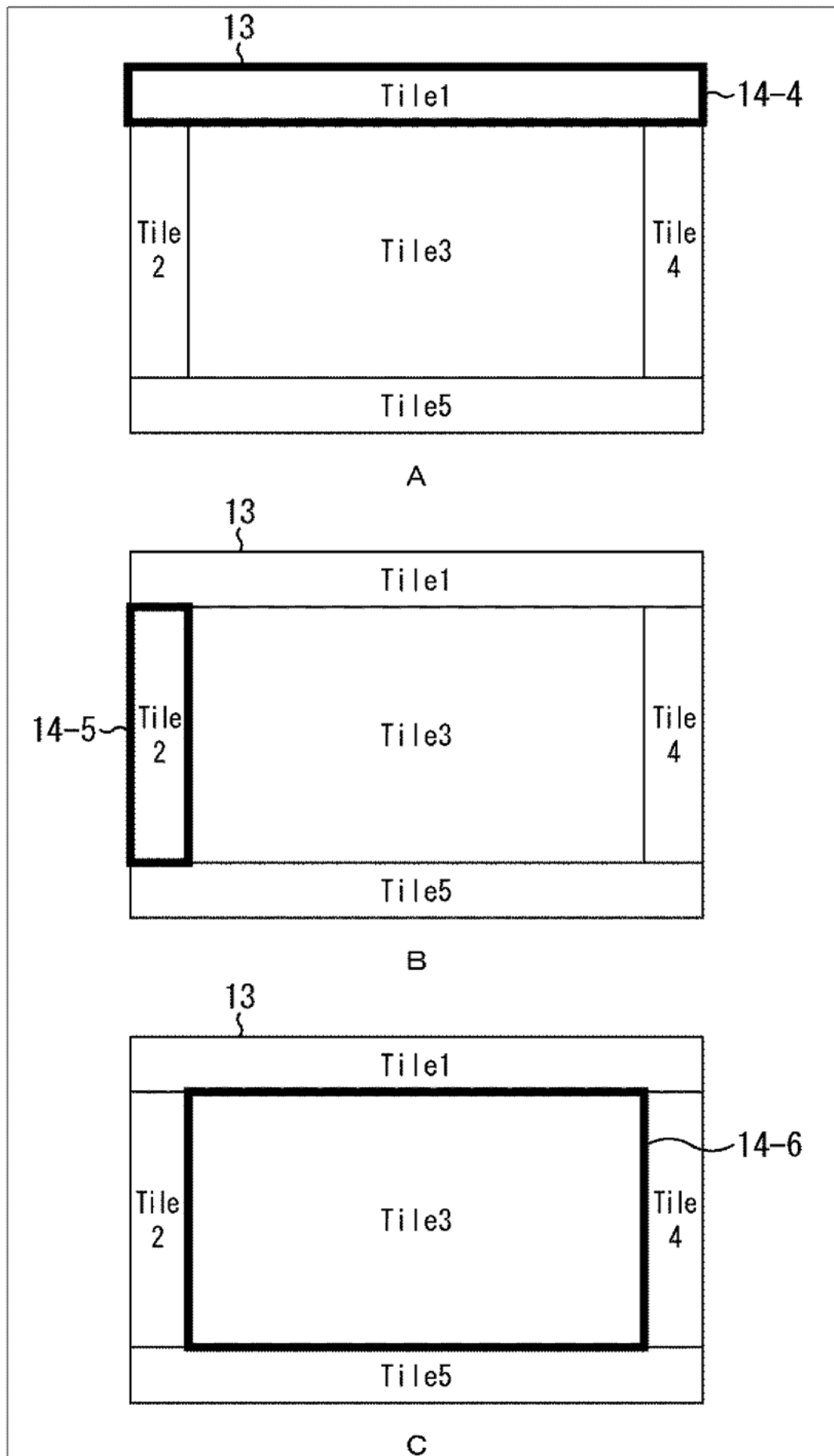


FIG. 19

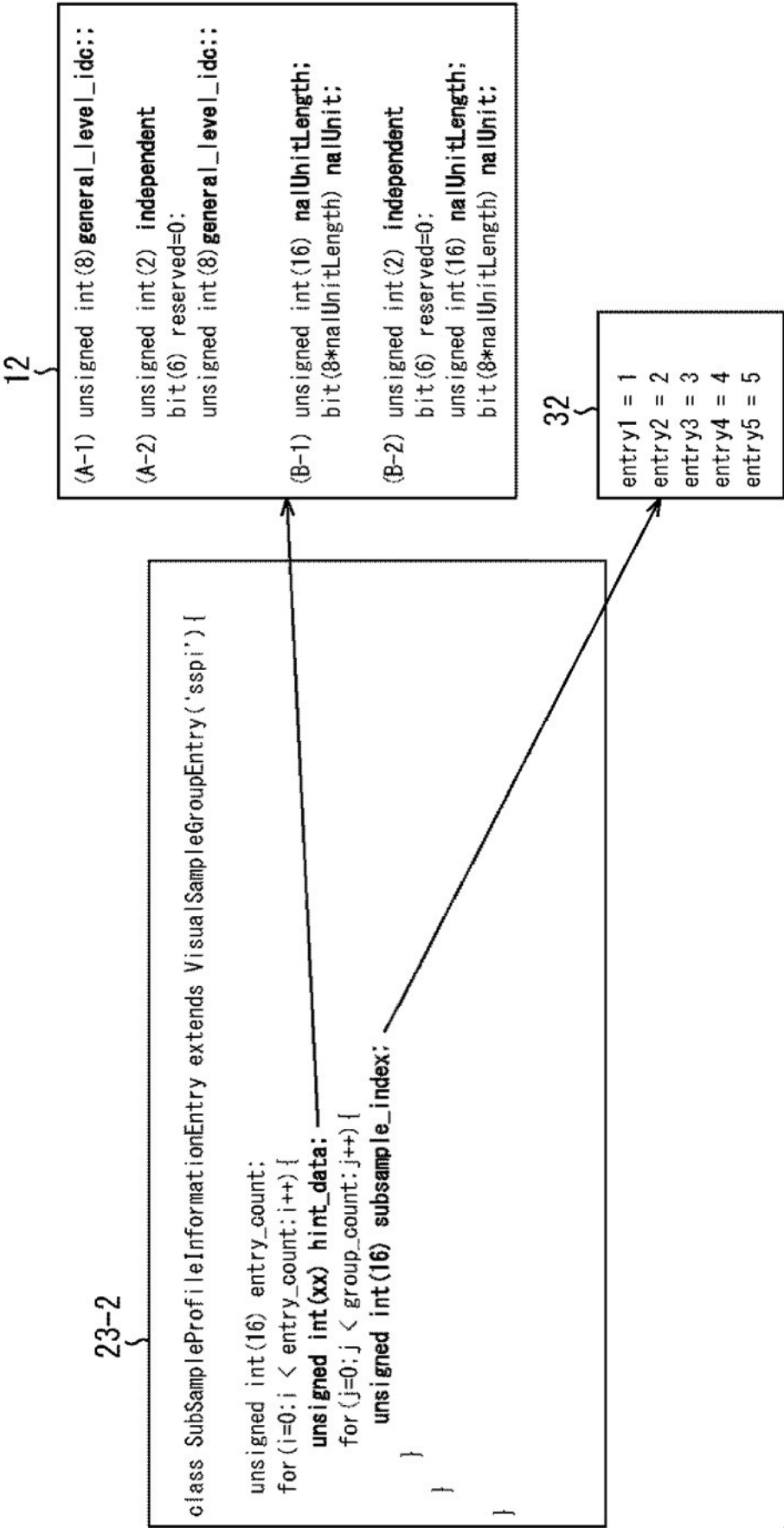


FIG. 20

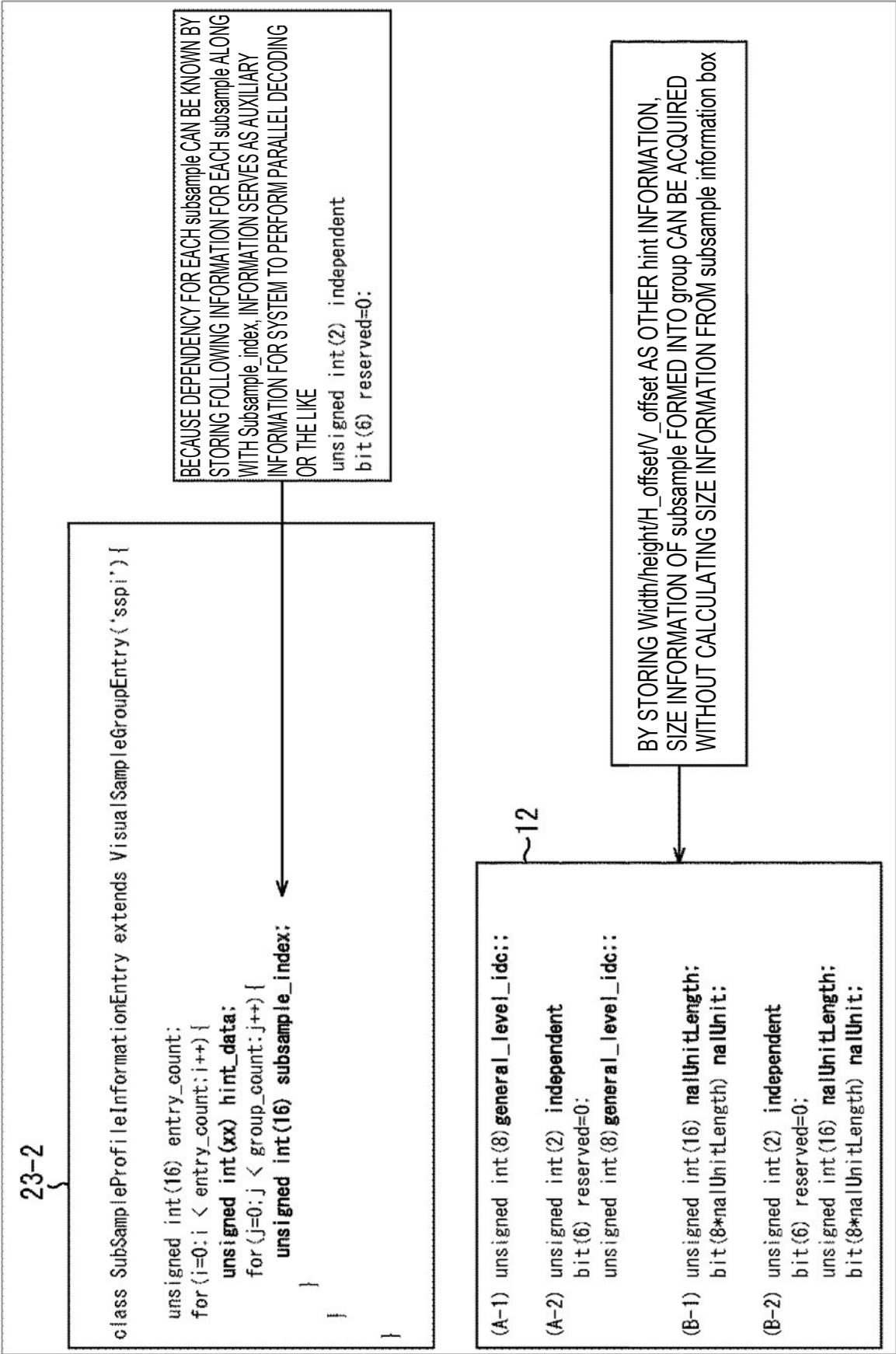
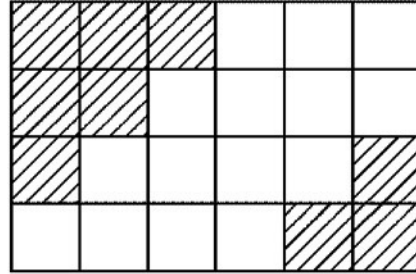
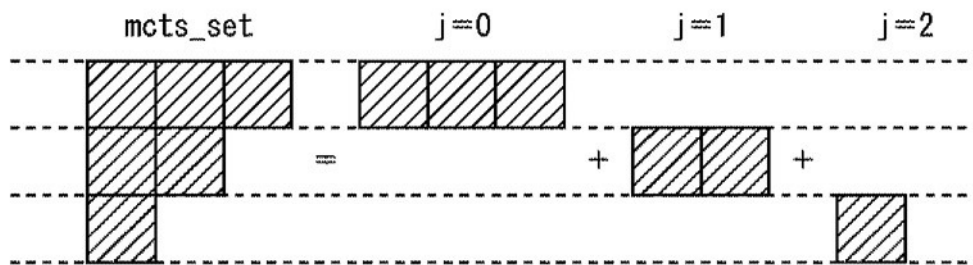
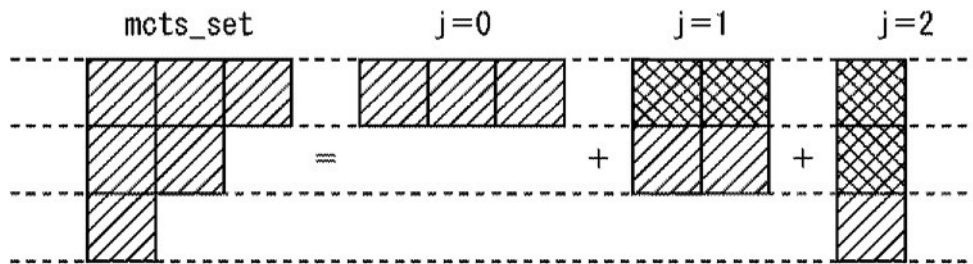


FIG. 21

A



B



C

$\text{CtbAddrTsToRs}[\text{top_left_tile_index}]$
 $\text{CtbAddrTsToRs}[\text{bottom_right_tile_index}]$
 $S_i = (X_{\text{right}, i} - X_{\text{left}, i}) * (Y_{\text{top}, i} - Y_{\text{btm}, i})$
 $S_{\text{sum}} = \sum S_i - \text{Overlap}$
 $\text{NumOfPixelToDecode} = S_{\text{sum}} * \text{CTUSize}$

D

FIG. 22

motion_constrained_tile_sets(payloadSize) {	Descriptor
num_sets_in_message_minus1	ue(v)
for (i=0; i<=num_sets_in_message_minus1; i++) {	
mcts_id[i]	ue(v)
num_tile_rects_in_set_minus1[i]	ue(v)
for (j=0; j<= num_tile_rects_in_set_minus1[i];	
top_left_tile_index[i][j]	ue(v)
bottom_right_tile_index[i][j]	ue(v)
}	
exact_sample_value_match_flag[i]	u(1)
}	
}	

FIG. 23

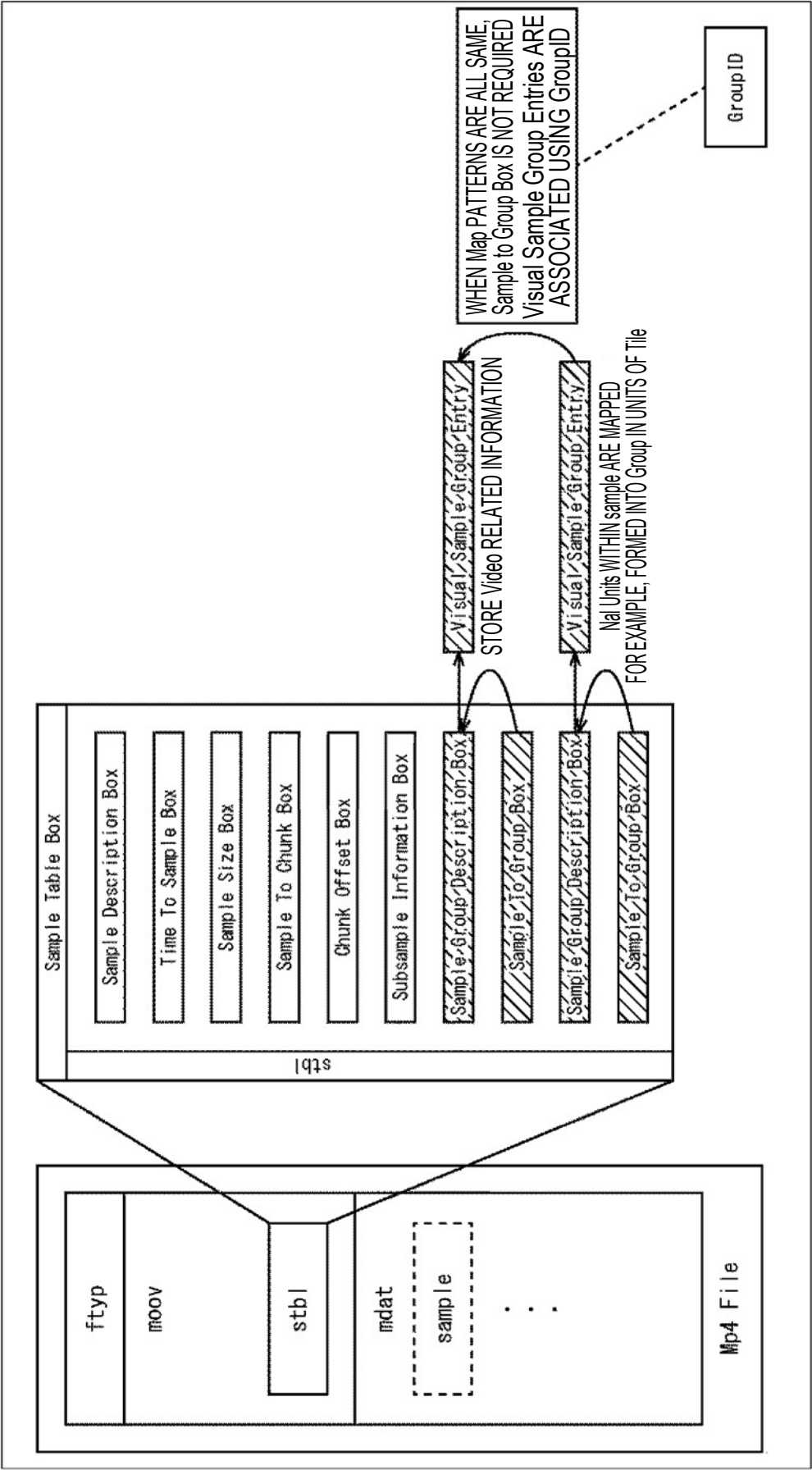


FIG. 24

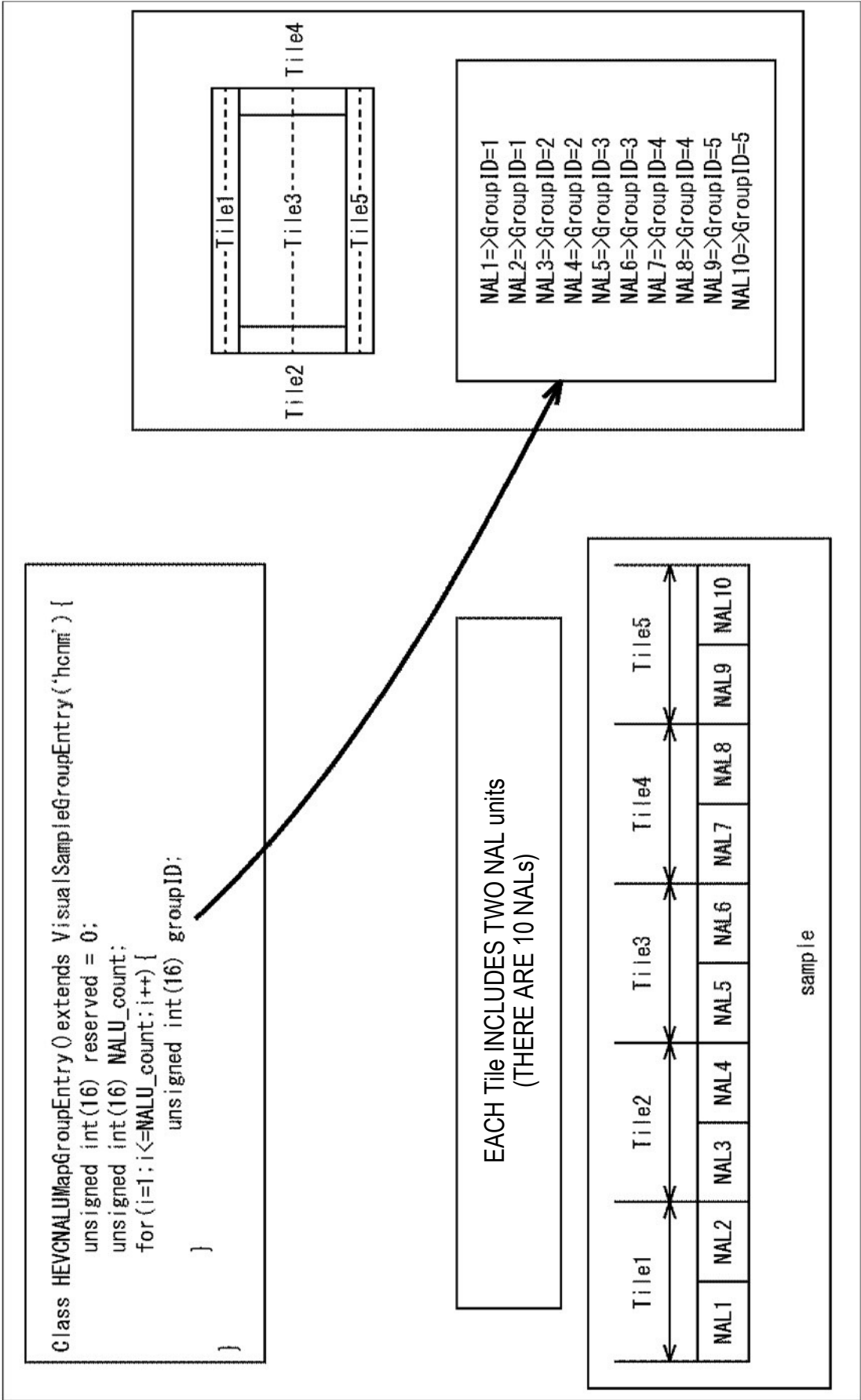


FIG. 25

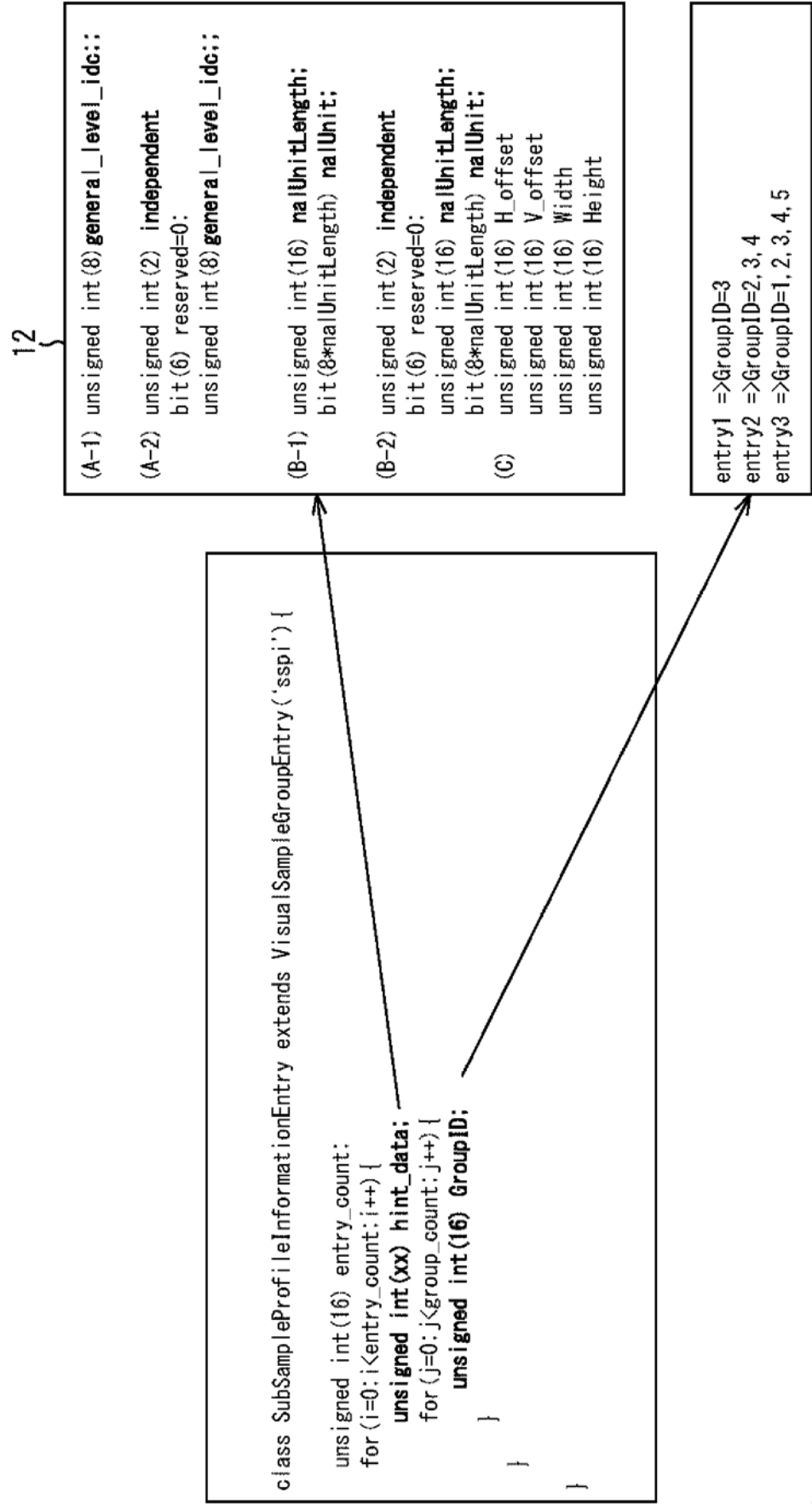


FIG. 26

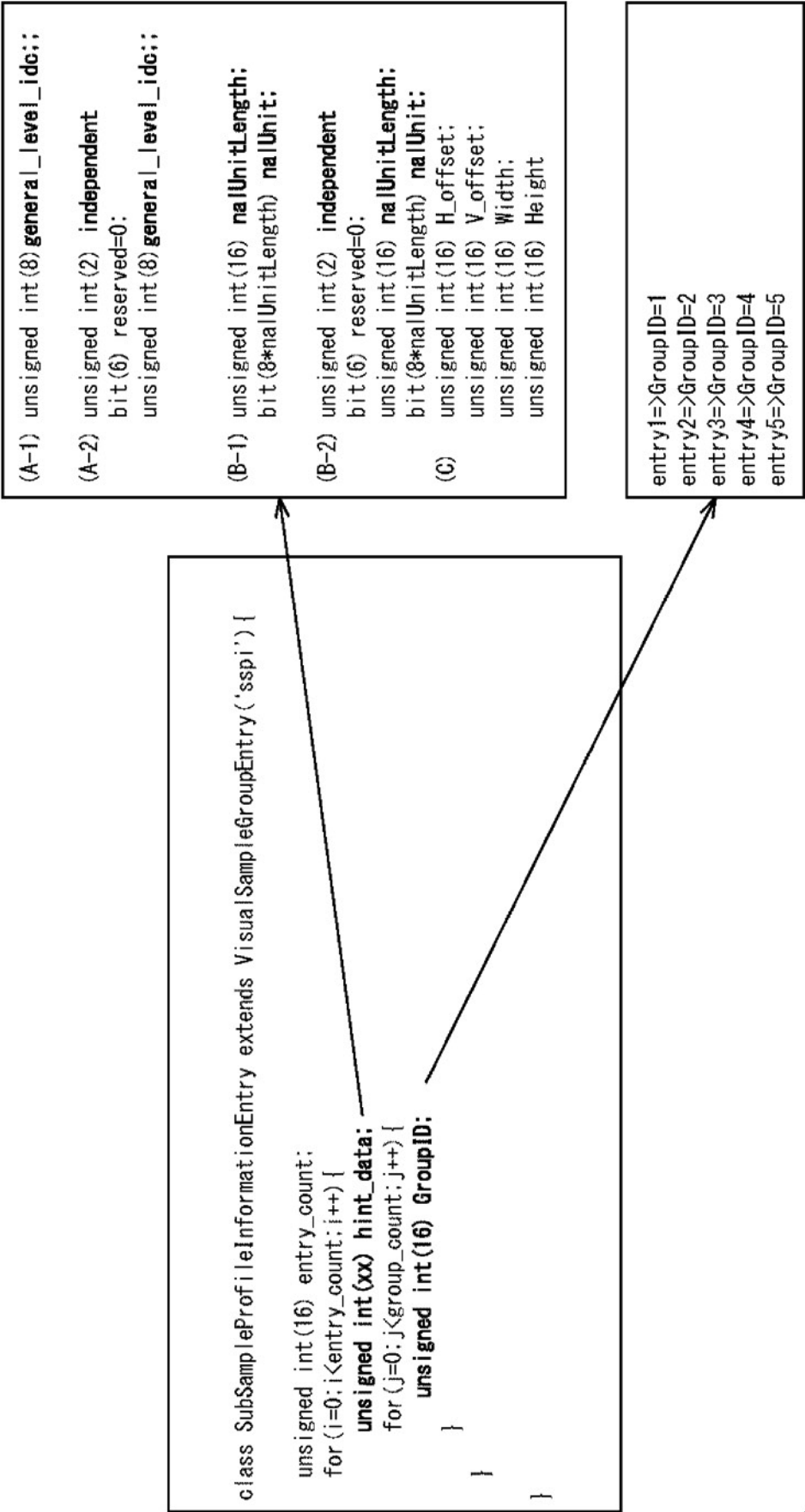


FIG. 27

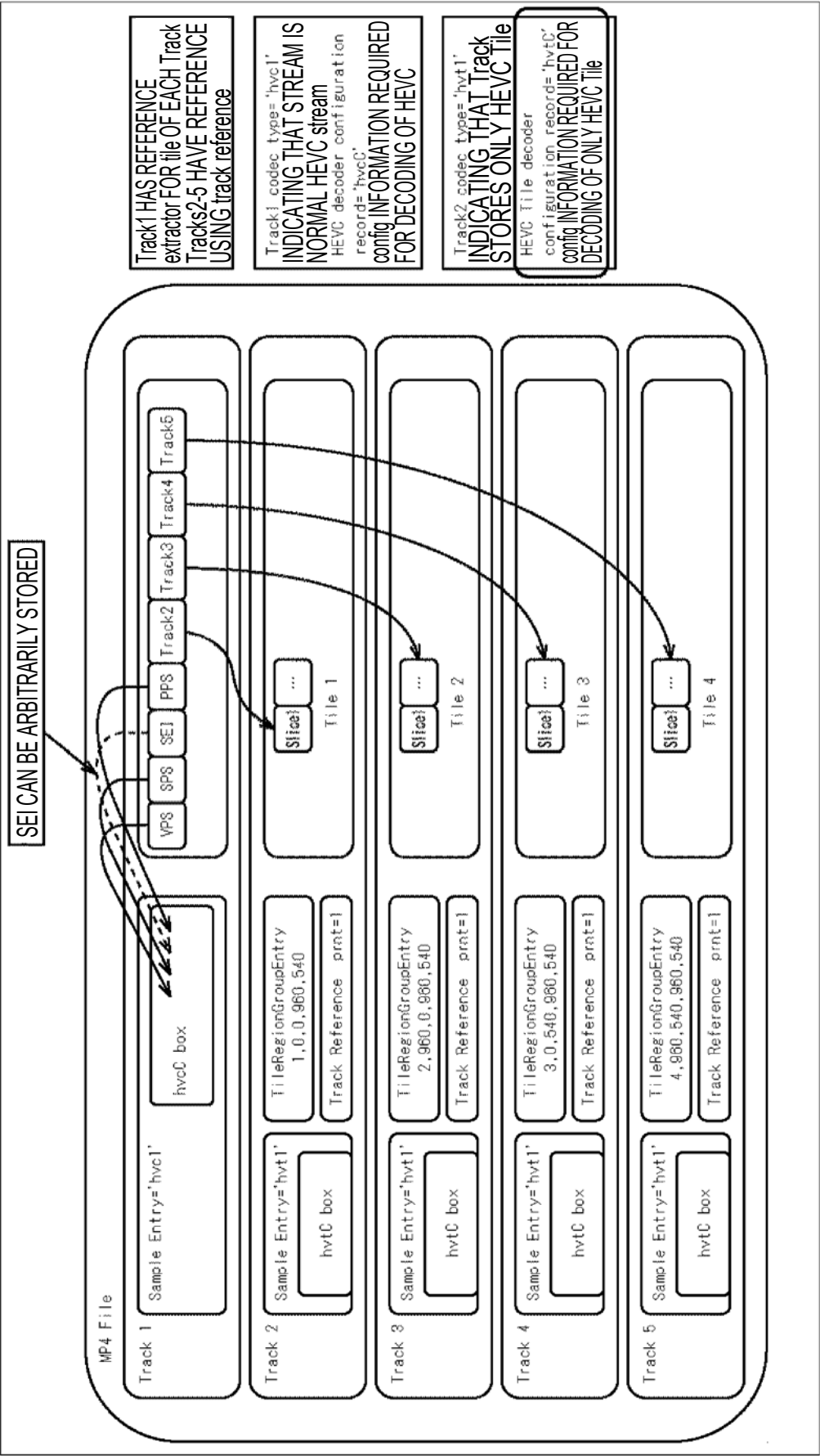


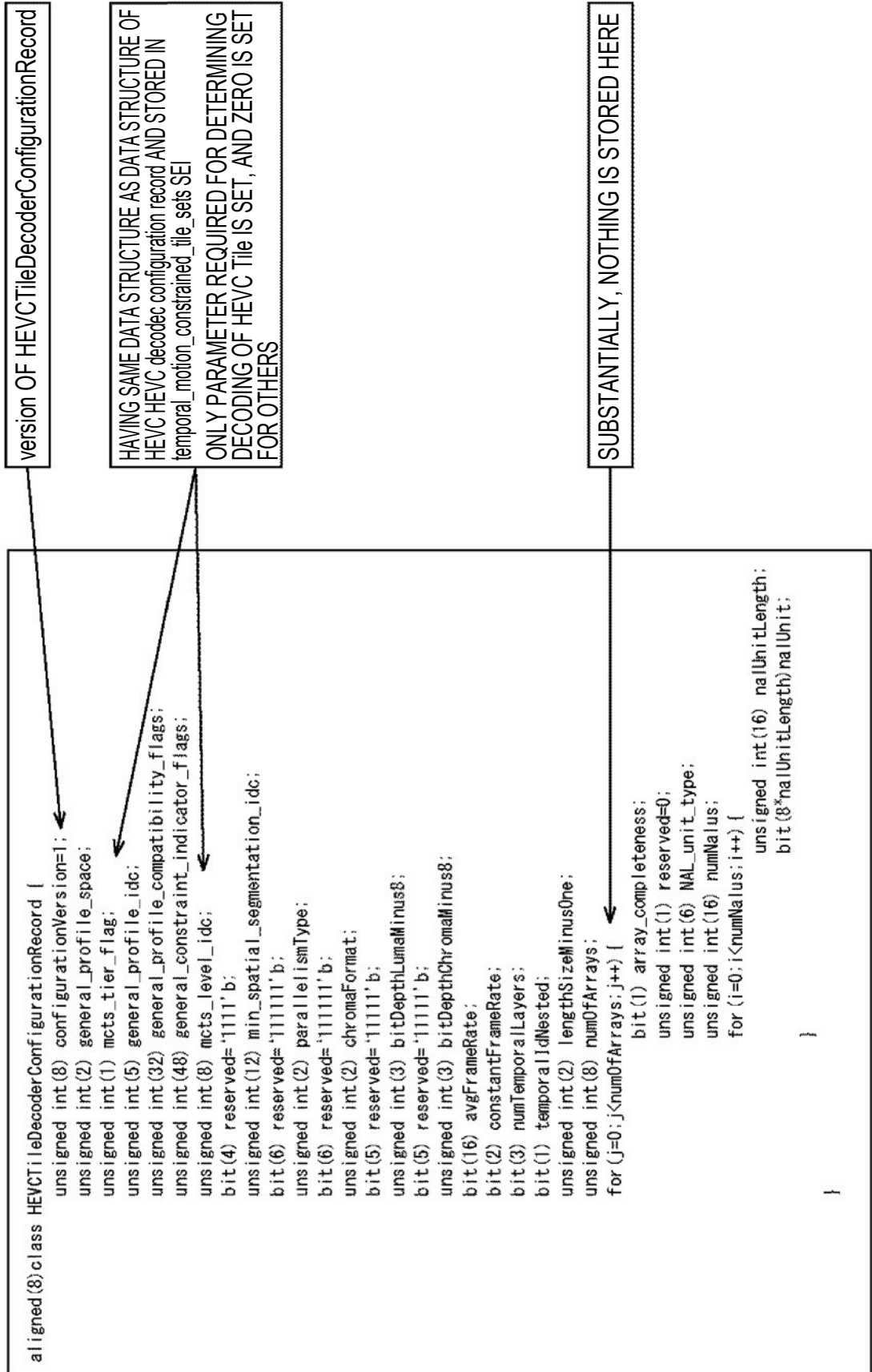
FIG. 28

FIG. 29

temporal_motion_constrained_tile_sets(payloadSize) {	Descriptor
mc_all_tiles_exact_sample_value_match_flag	u(1)
each_tile_one_tile_set_flag	u(1)
limited_tile_set_display_flag	u(1)
if(!each_tile_one_tile_set_flag) {	
num_sets_in_message_minus1	ue(v)
for(i=0; i<= num_sets_in_message_minus1; i++) {	
mcts_id[i]	ue(v)
if(limited_tile_set_display_flag)	
display_tile_set_flag[i]	u(1)
num_tile_rects_in_set_minus1[i]	ue(v)
for(j=0; j<= num_tile_rects_in_set_minus1[i]; j++) {	
top_left_tile_index[i][j]	ue(v)
bottom_right_tile_index[i][j]	ue(v)
}	
if(! mc_all_tiles_exact_sample_value_match_flag)	
exact_sample_value_match_flag[i]	u(1)
mcts_tier_level_idc_present_flag[i]	u(1)
if(mcts_tier_level_idc_present_flag[i]) {	
mcts_tier_flag[i]	u(1)
mcts_level_idc[i]	u(8)
}	
}	
} else {	
max_mcts_tier_level_idc_present_flag	u(1)
if(max_mcts_tier_level_idc_present_flag) {	
max_mcts_tier_flag	u(1)
max_mcts_level_idc	u(8)
}	
}	
}	

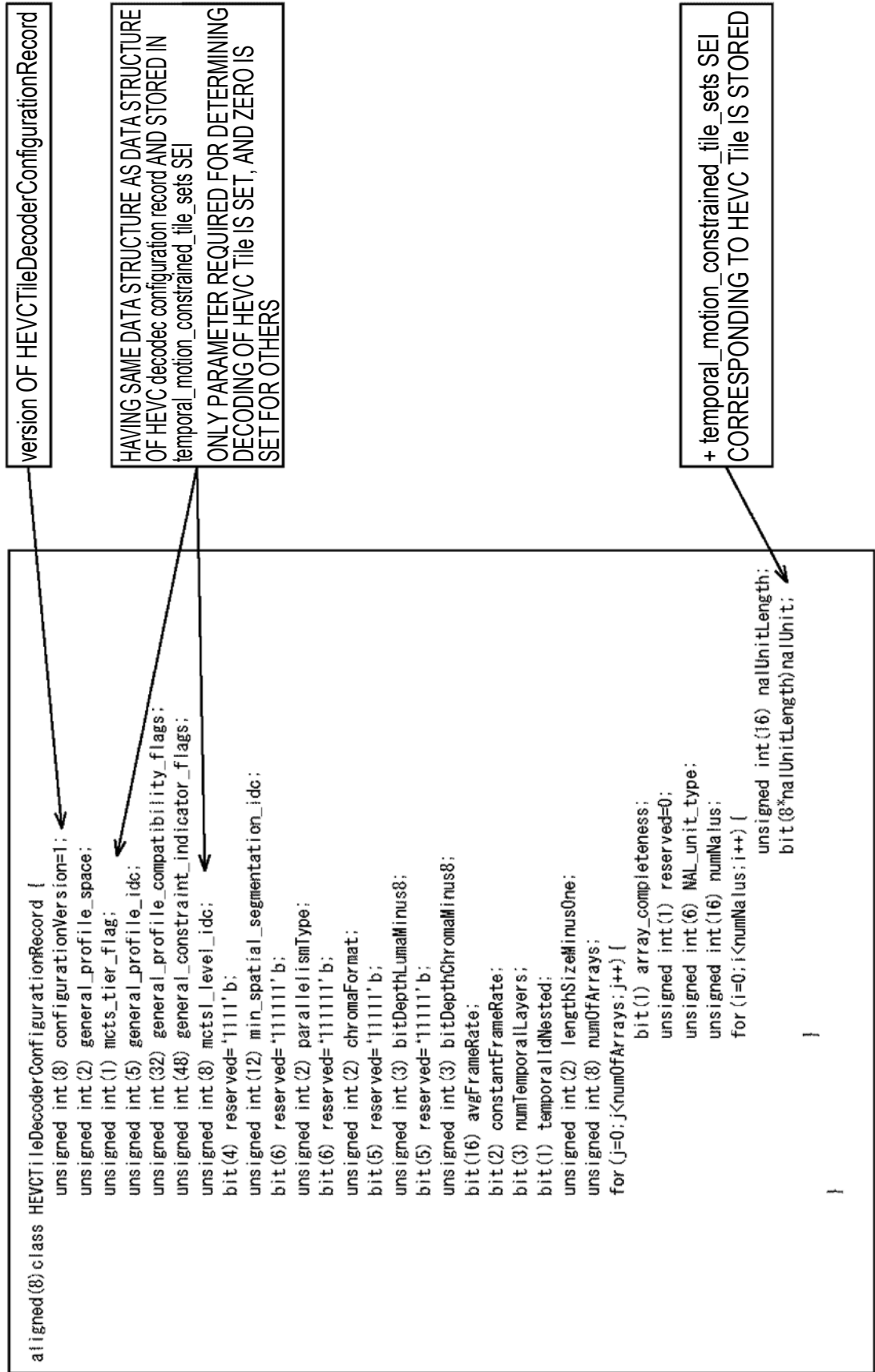
FIG. 30

FIG. 31

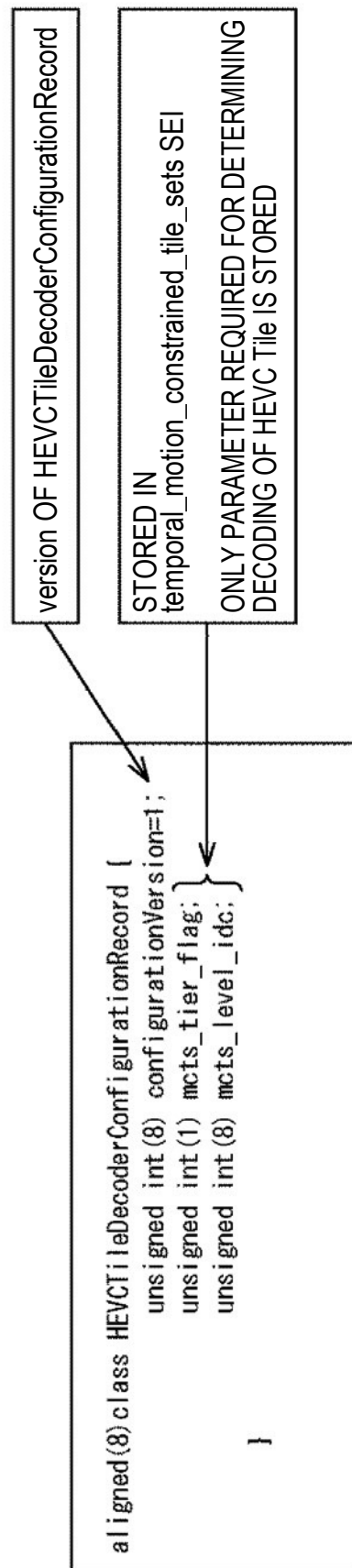


FIG. 32

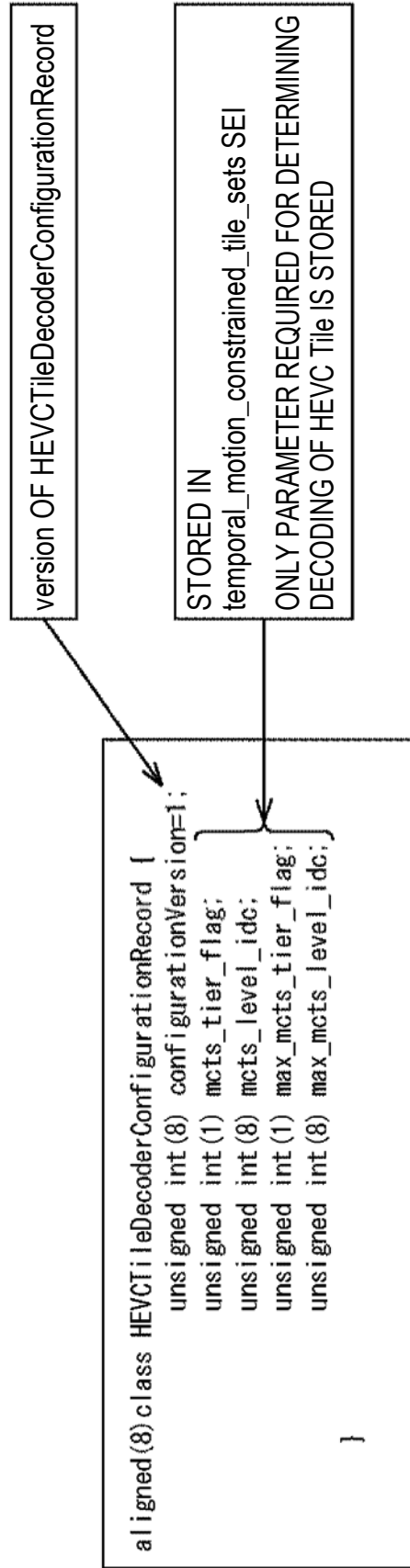


FIG. 33

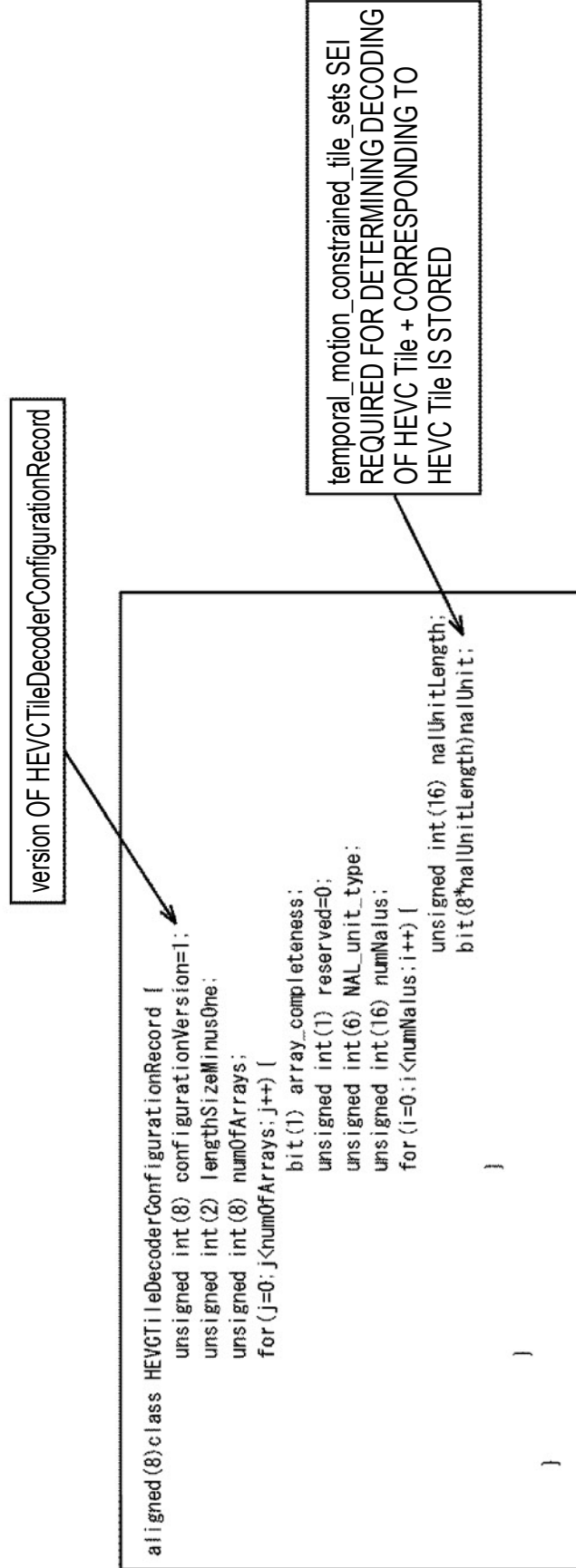


FIG. 34

version OF HEVC TileDecoderConfigurationRecord

```

aligned(8) class HEVC TileDecoderConfigurationRecord {
    unsigned int(8) configurationVersion=1;
    unsigned int(1) mcts_tier_flag;
    unsigned int(8) mcts_level_idc;
    unsigned int(2) lengthSizeMinusOne;
    unsigned int(8) numOfArrays;
    for (j=0; j<numOfArrays; j++) {
        bit(1) array_completeness;
        unsigned int(1) reserved=0;
        unsigned int(6) NAL_unit_type;
        unsigned int(16) numNalus;
        for (i=0; i<numNalus; i++) {
            unsigned int(16) nalUnitLength;
            bit(8*nalUnitLength) nalUnit;
        }
    }
}

```

STORED IN
temporal_motion_constrained_tile_sets SEI
ONLY PARAMETER REQUIRED FOR
DETERMINING DECODING OF HEVC Tile IS STORED

+ temporal_motion_constrained_tile_sets SEI
CORRESPONDING TO HEVC Tile IS STORED

FIG. 35

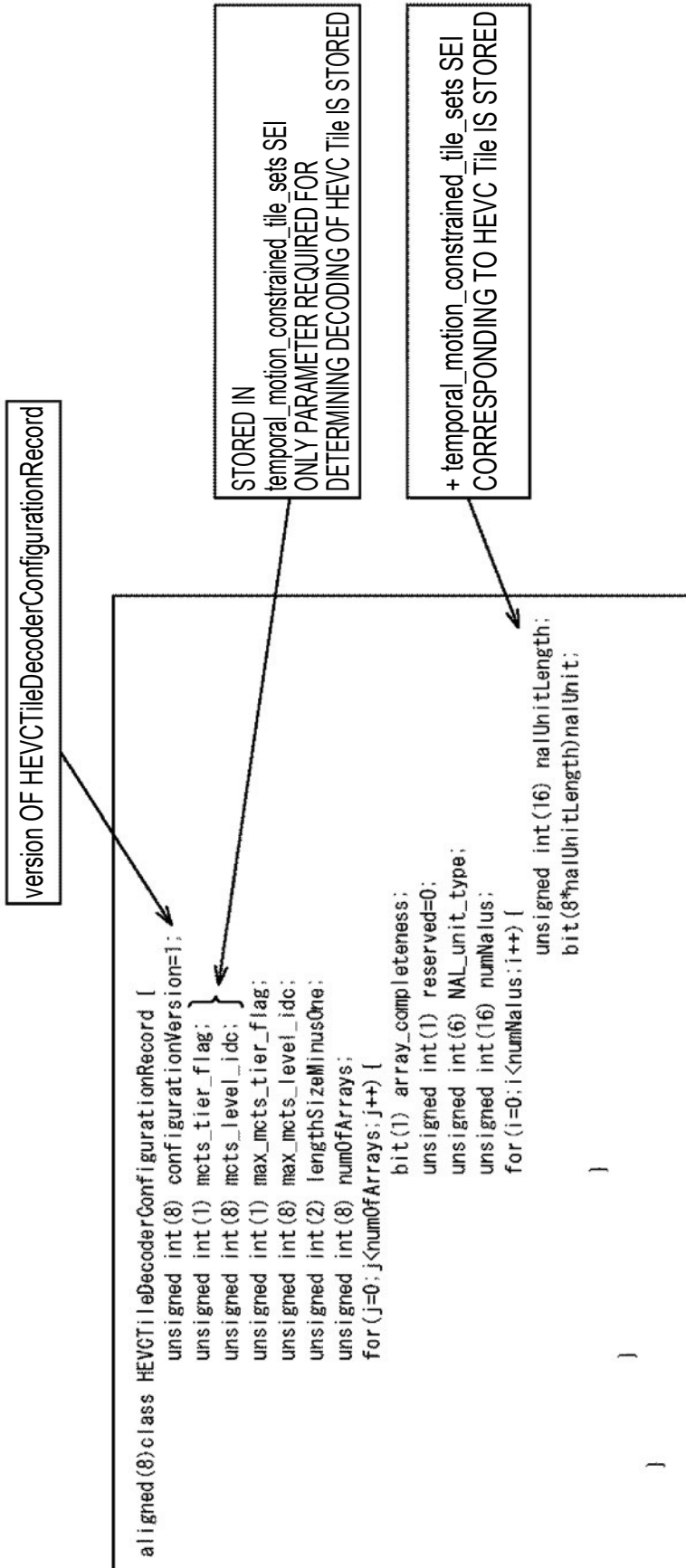


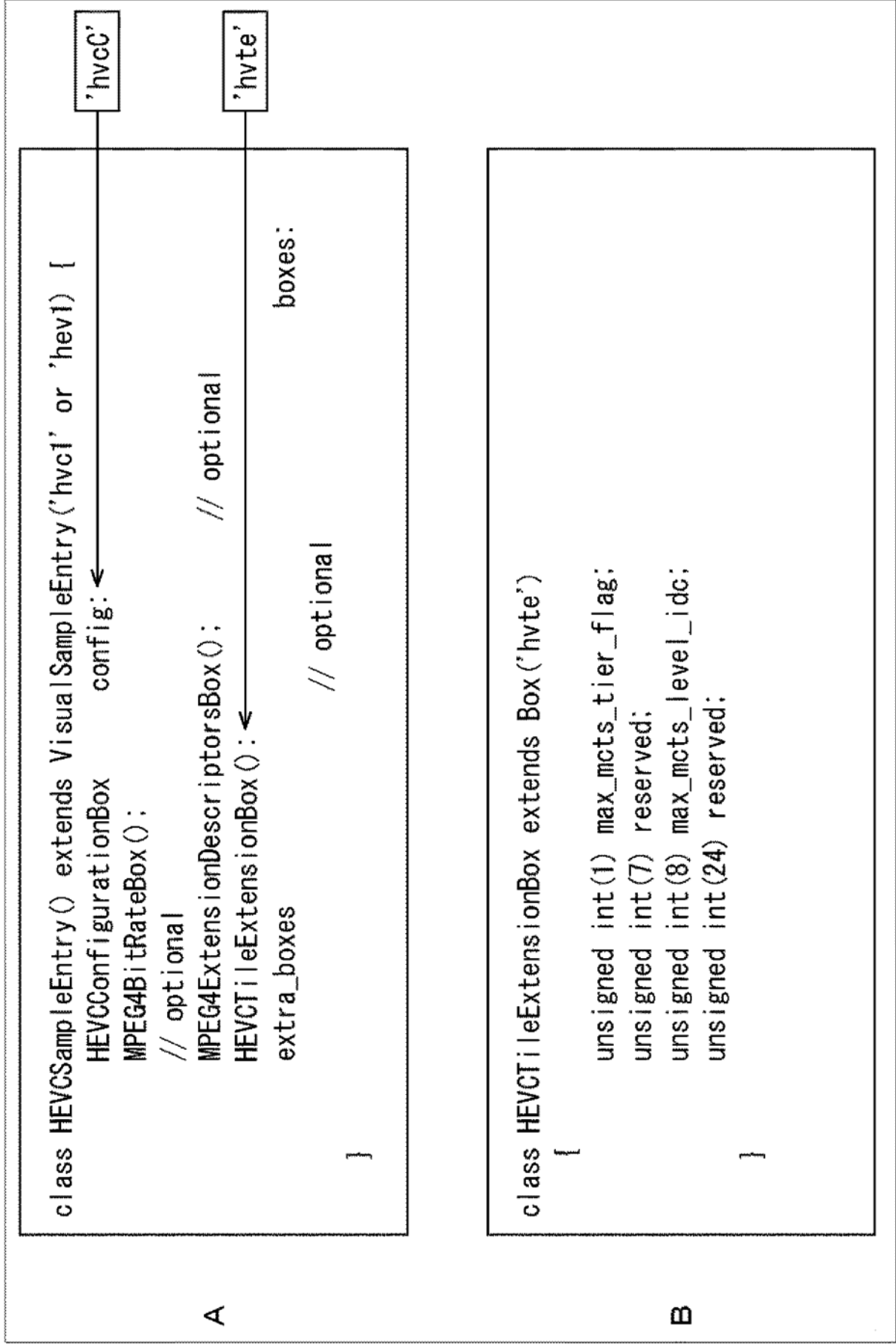
FIG. 37

FIG. 38

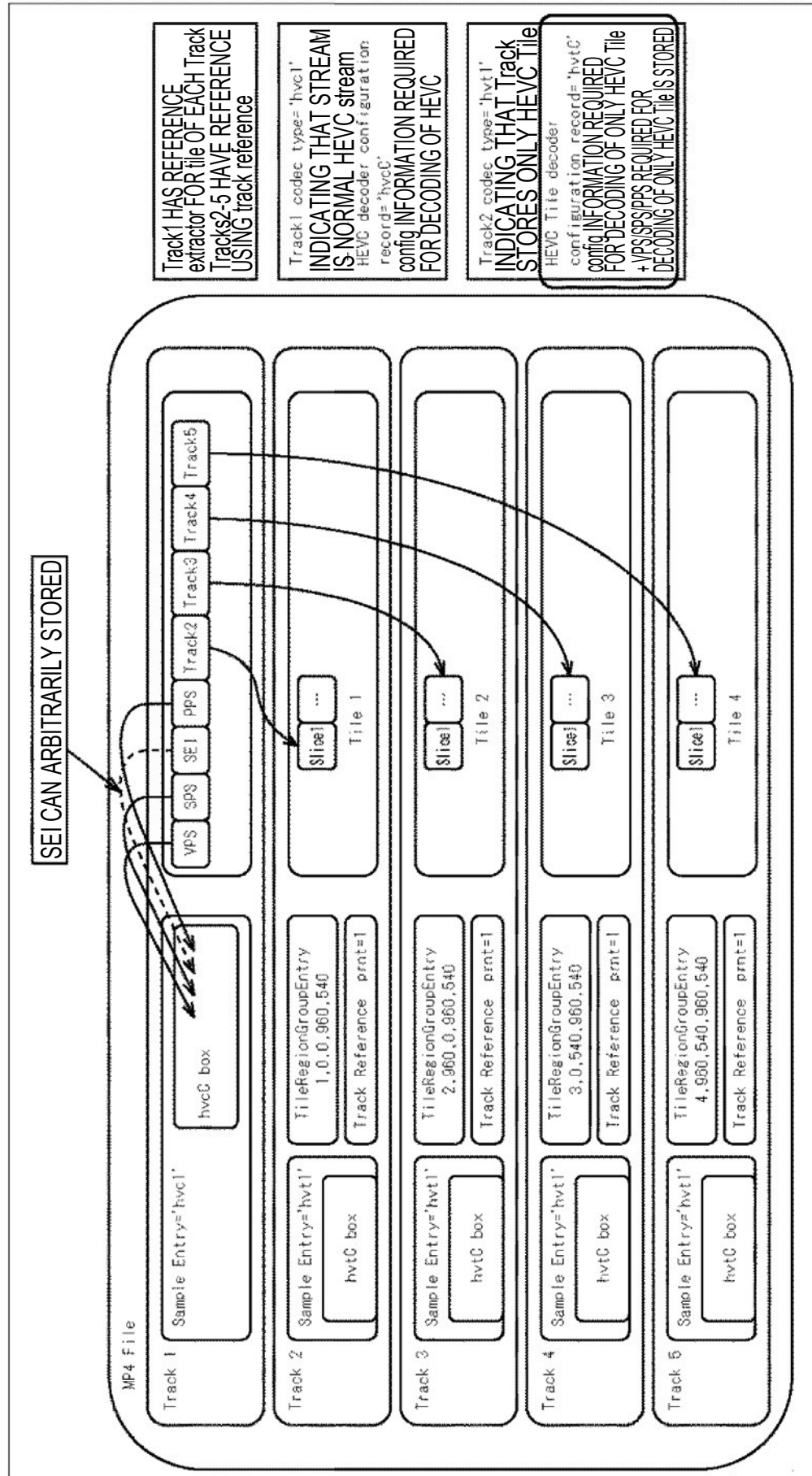


FIG. 39

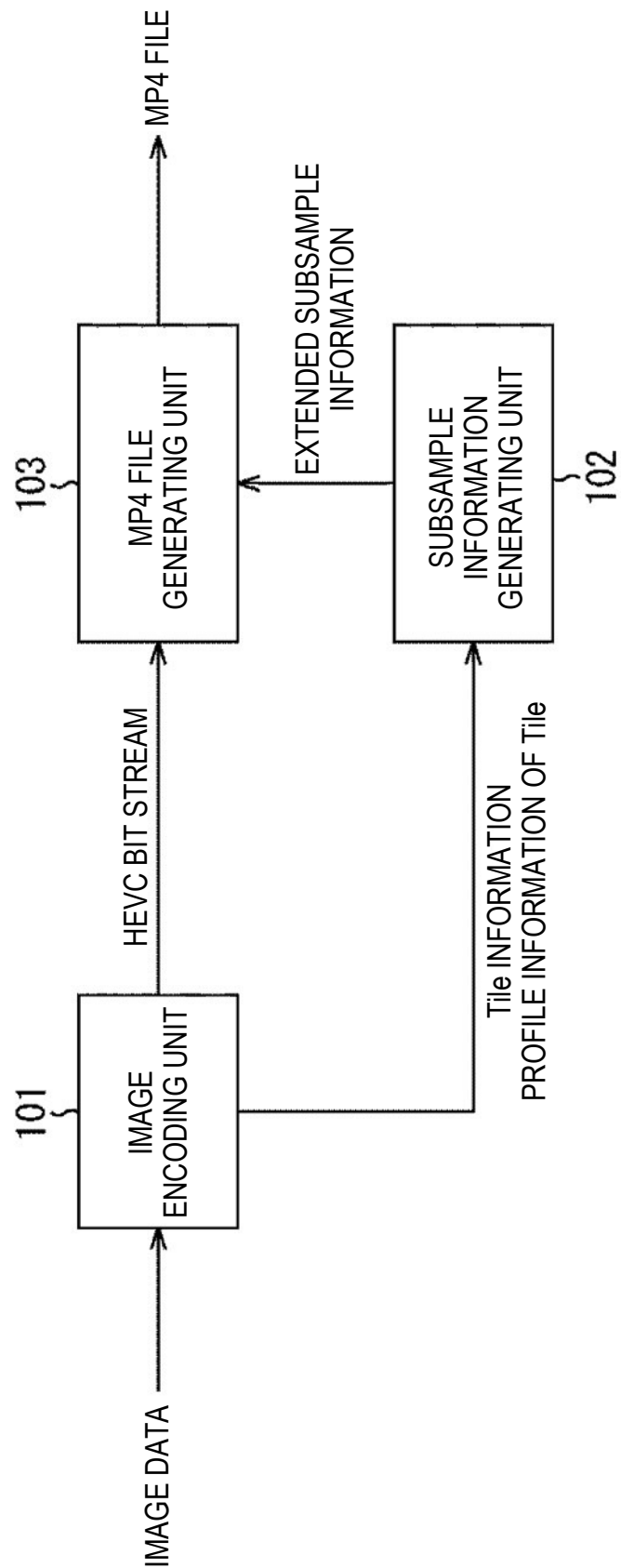


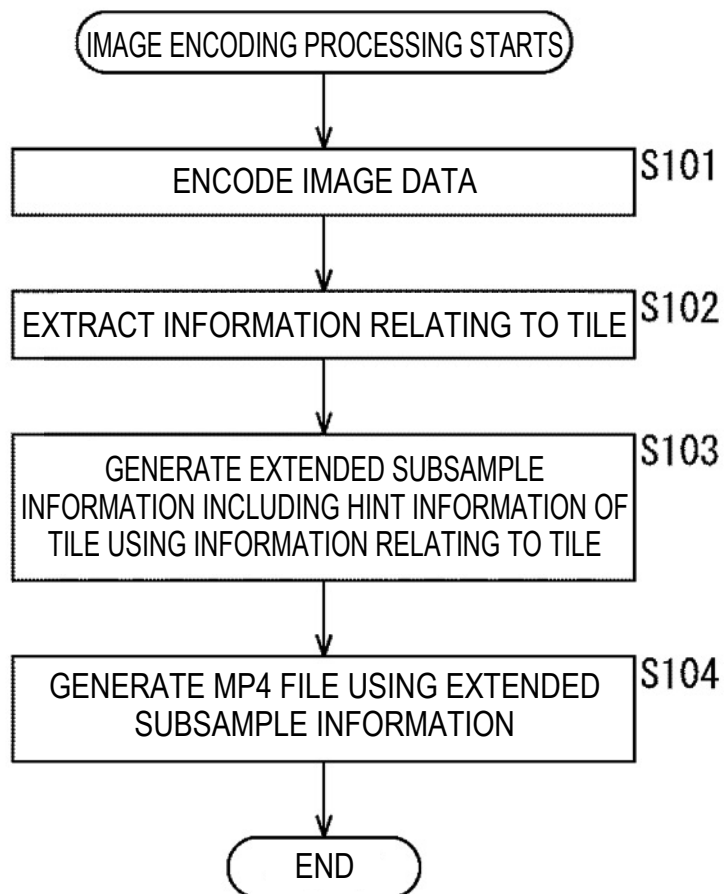
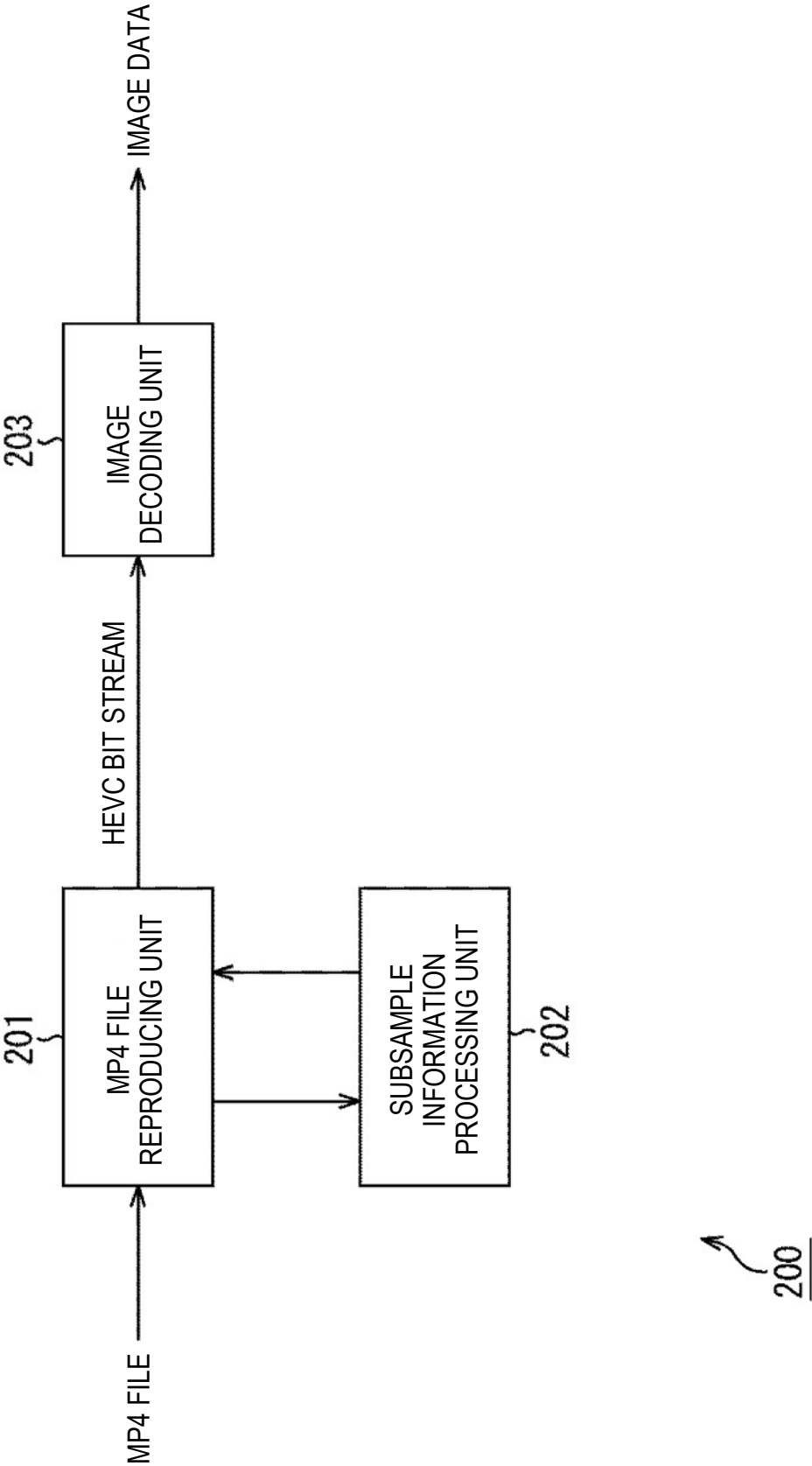
FIG. 40

FIG. 41



200 ↗

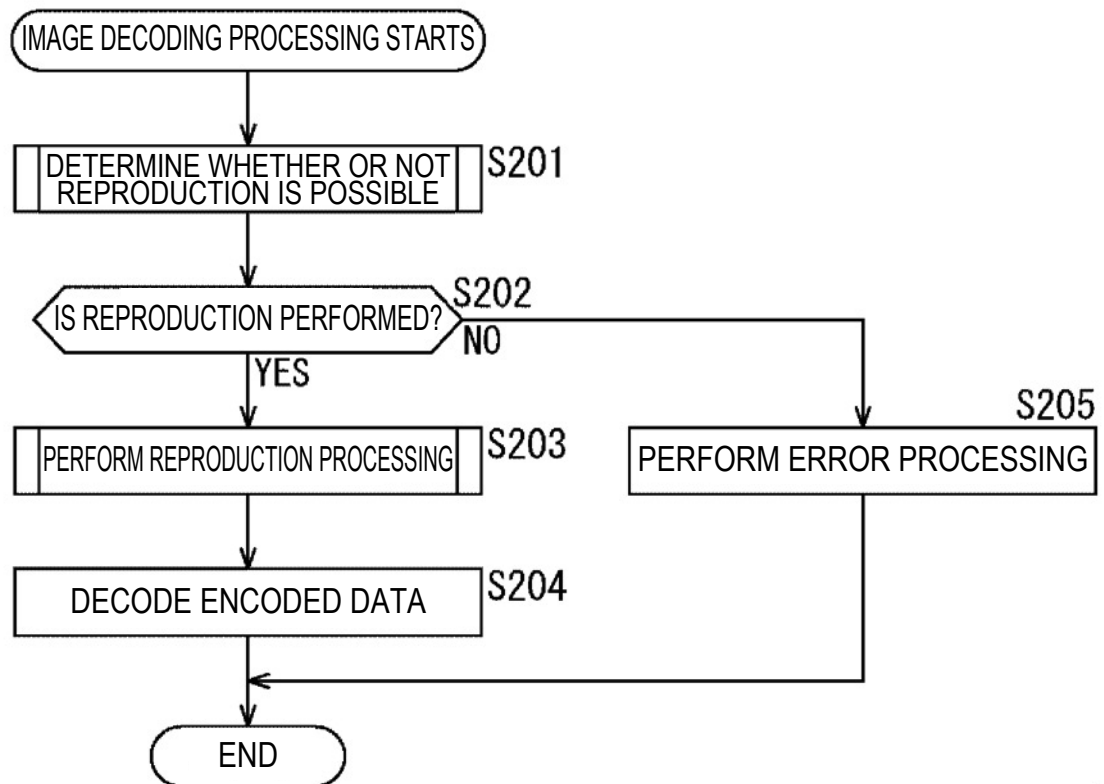
FIG. 42

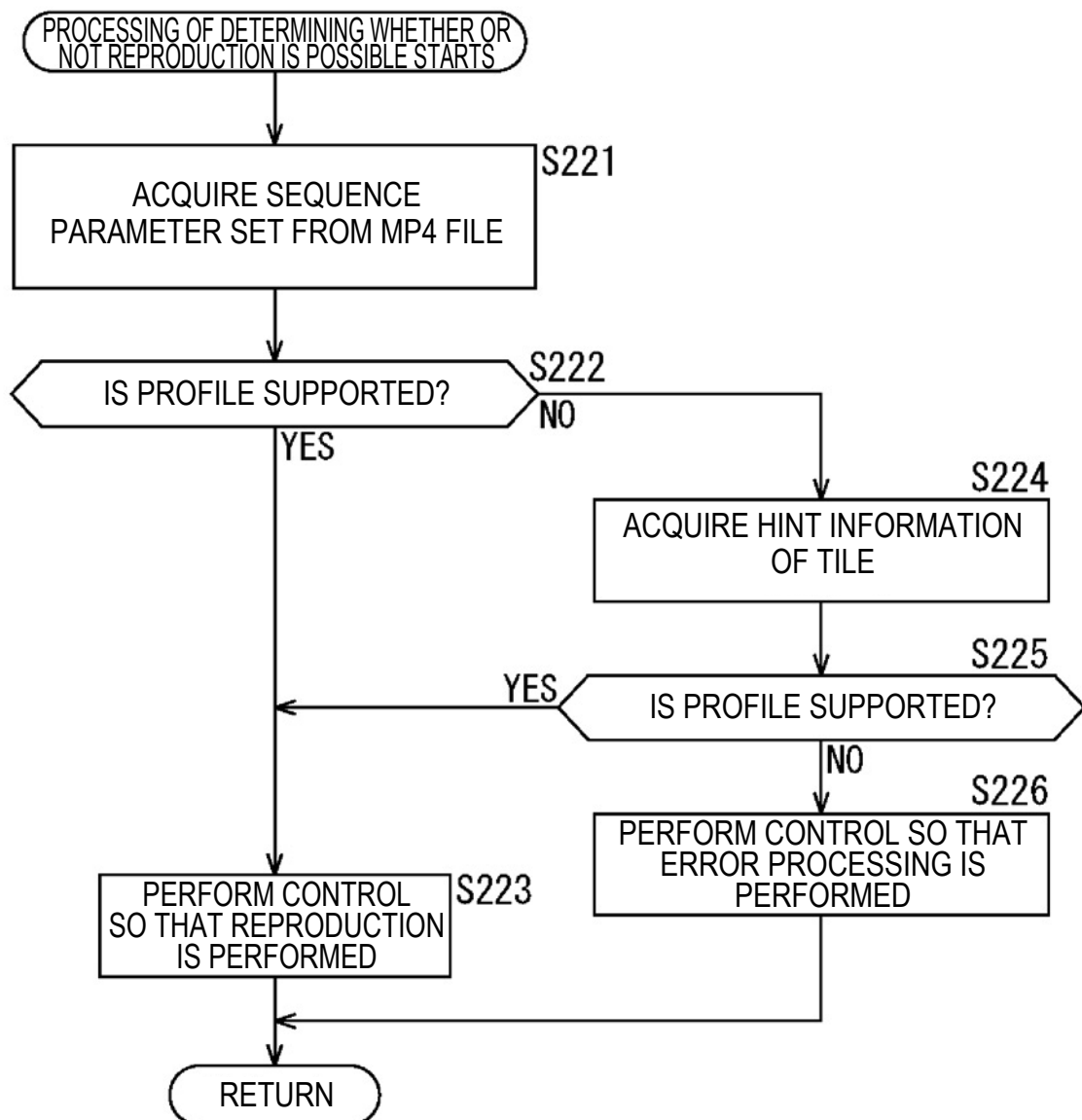
FIG. 43

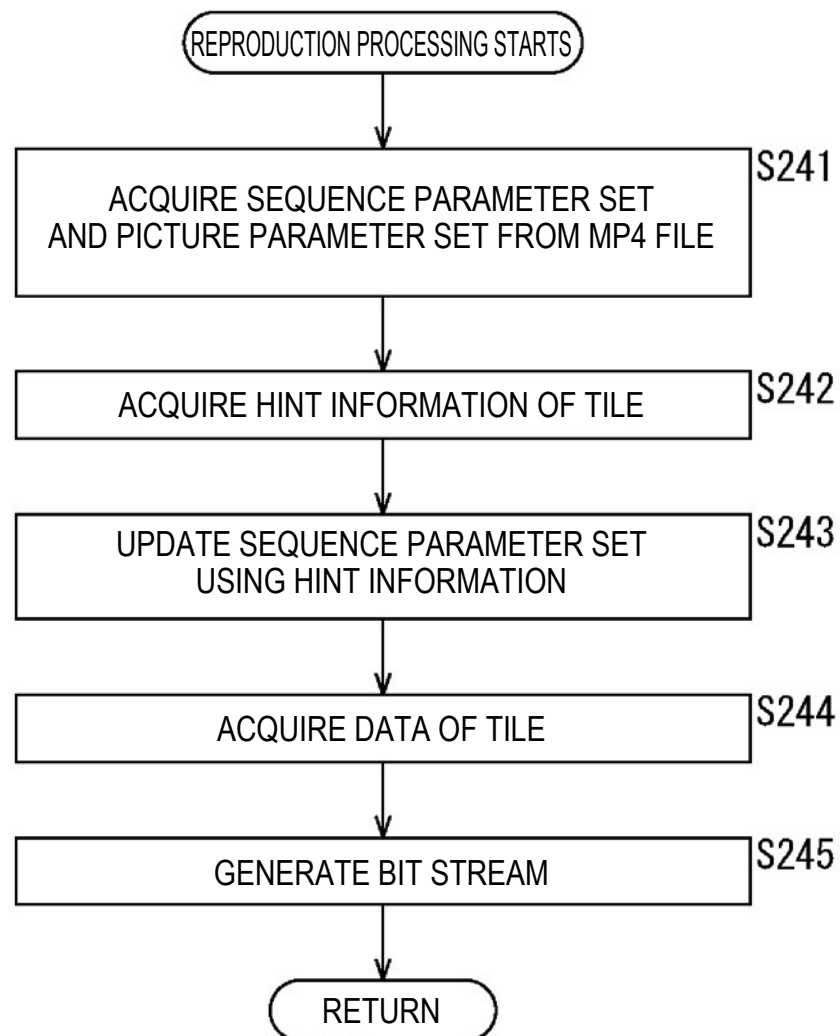
FIG. 44

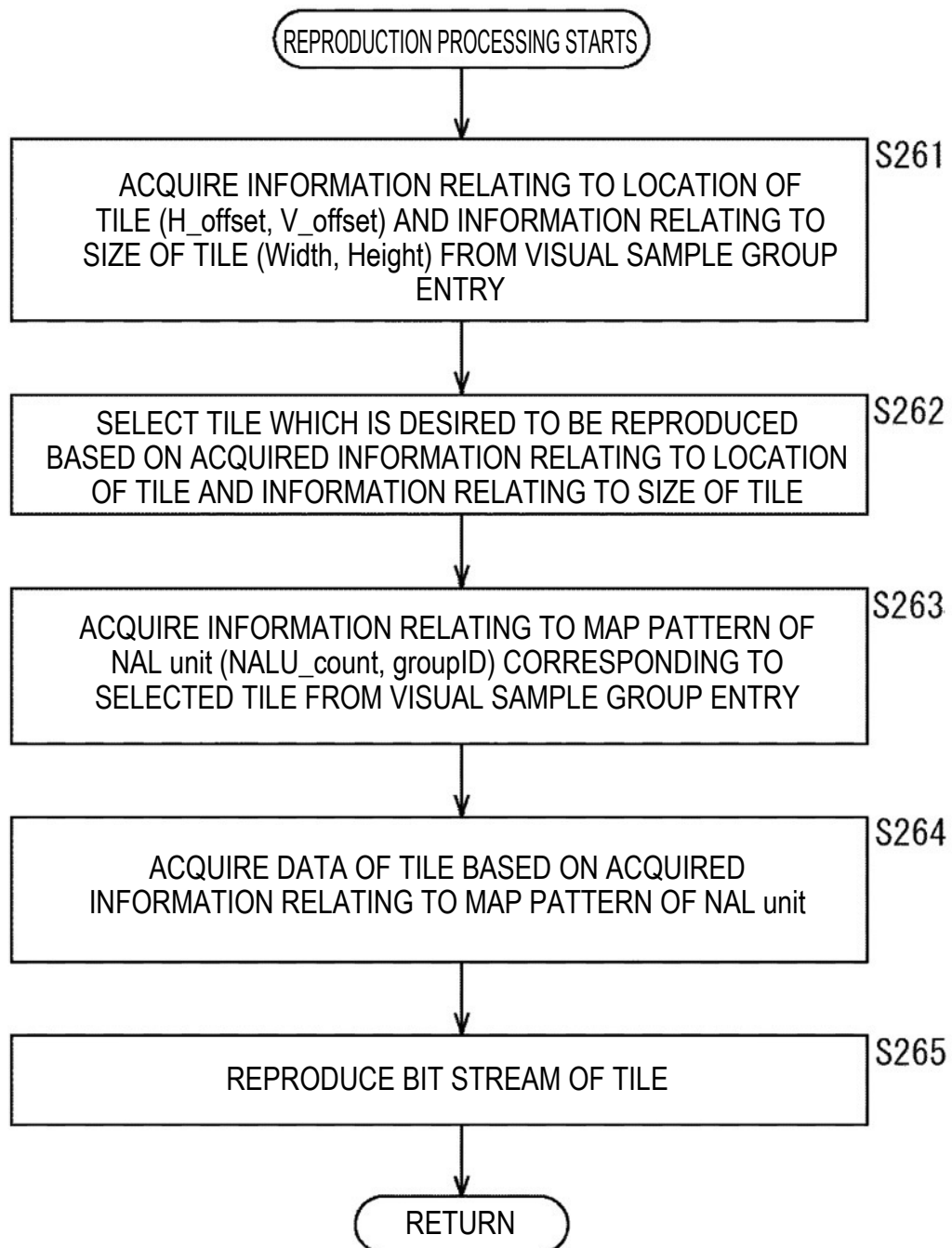
FIG. 45

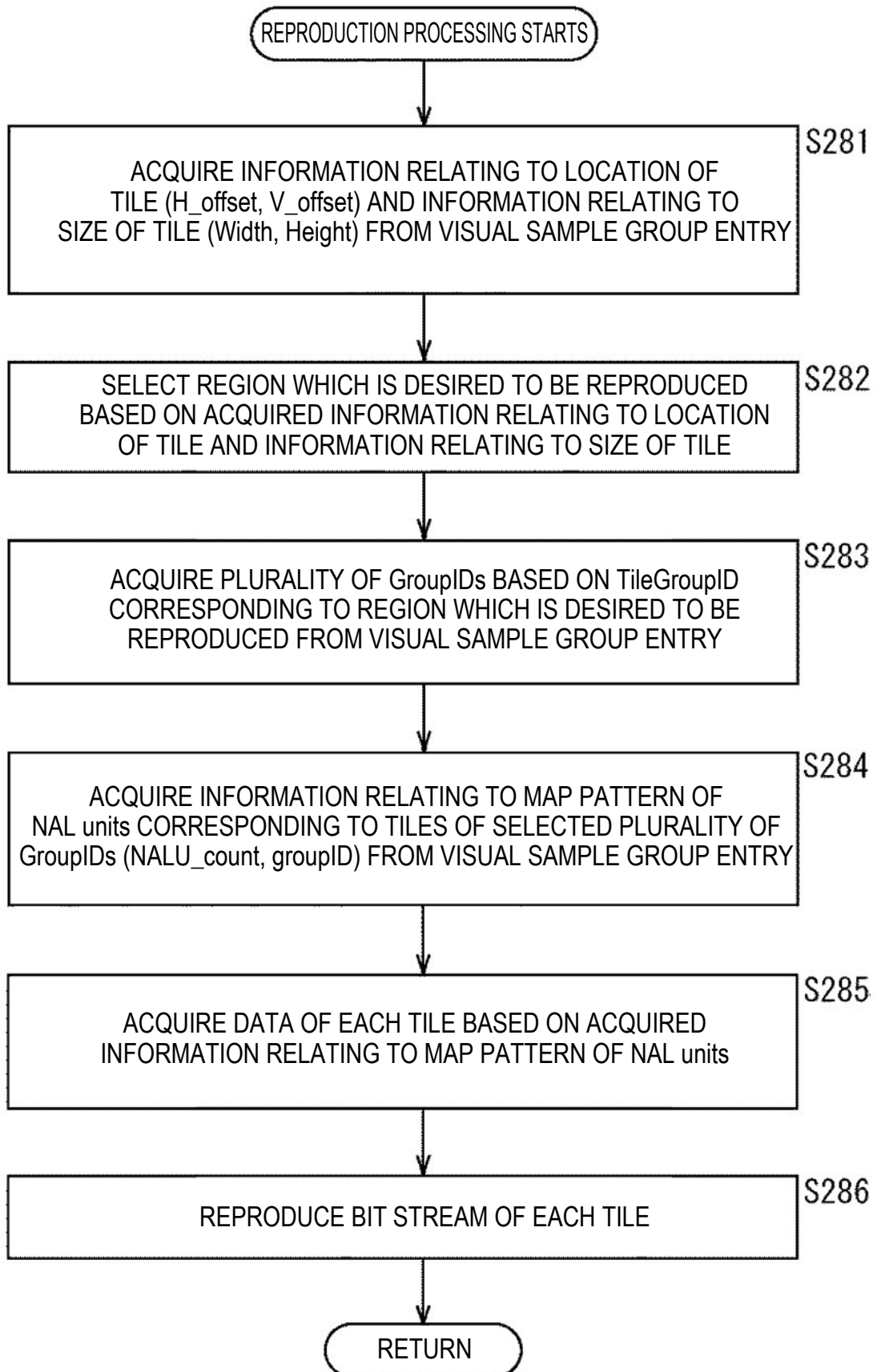
FIG. 46

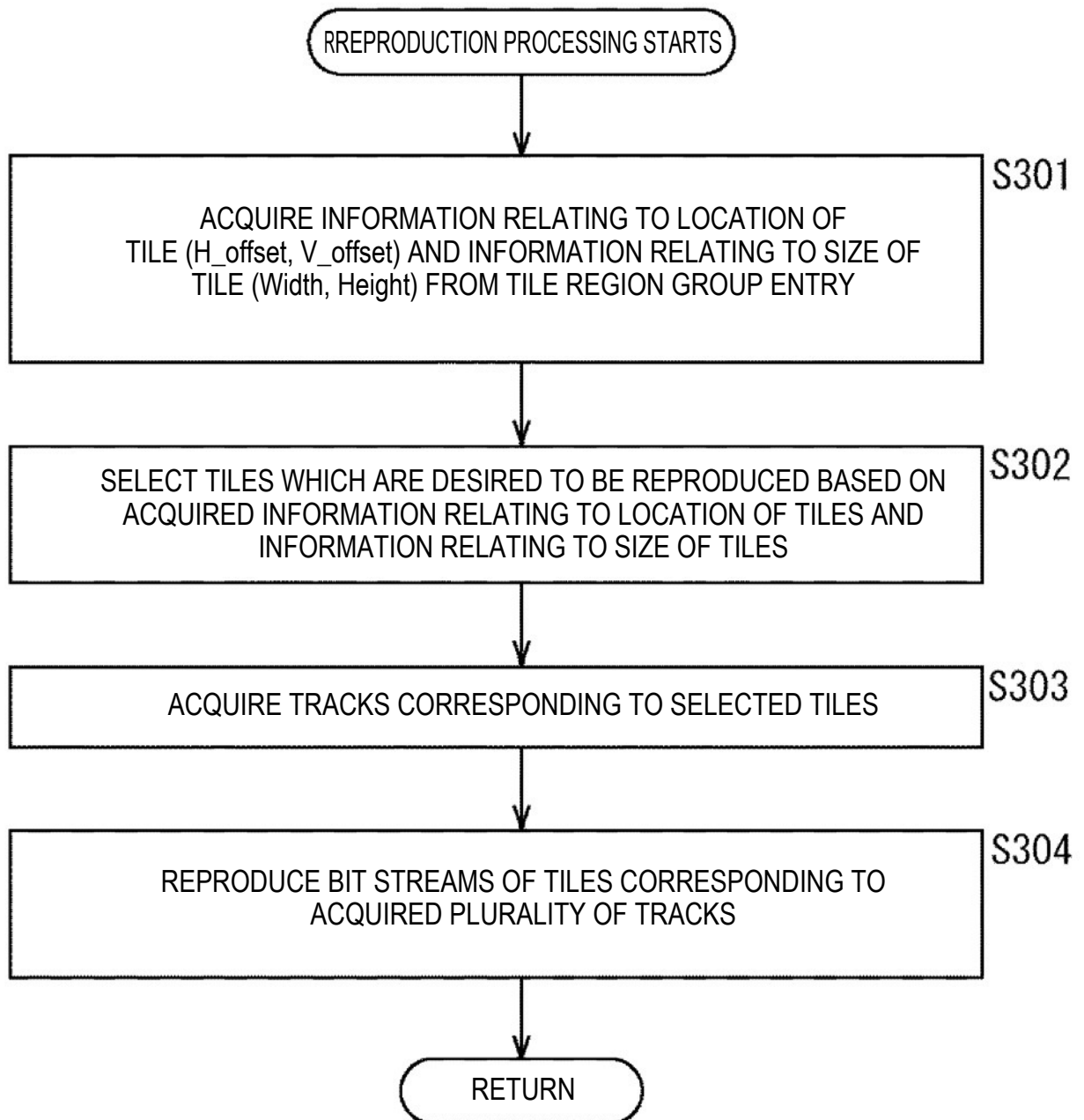
FIG. 47

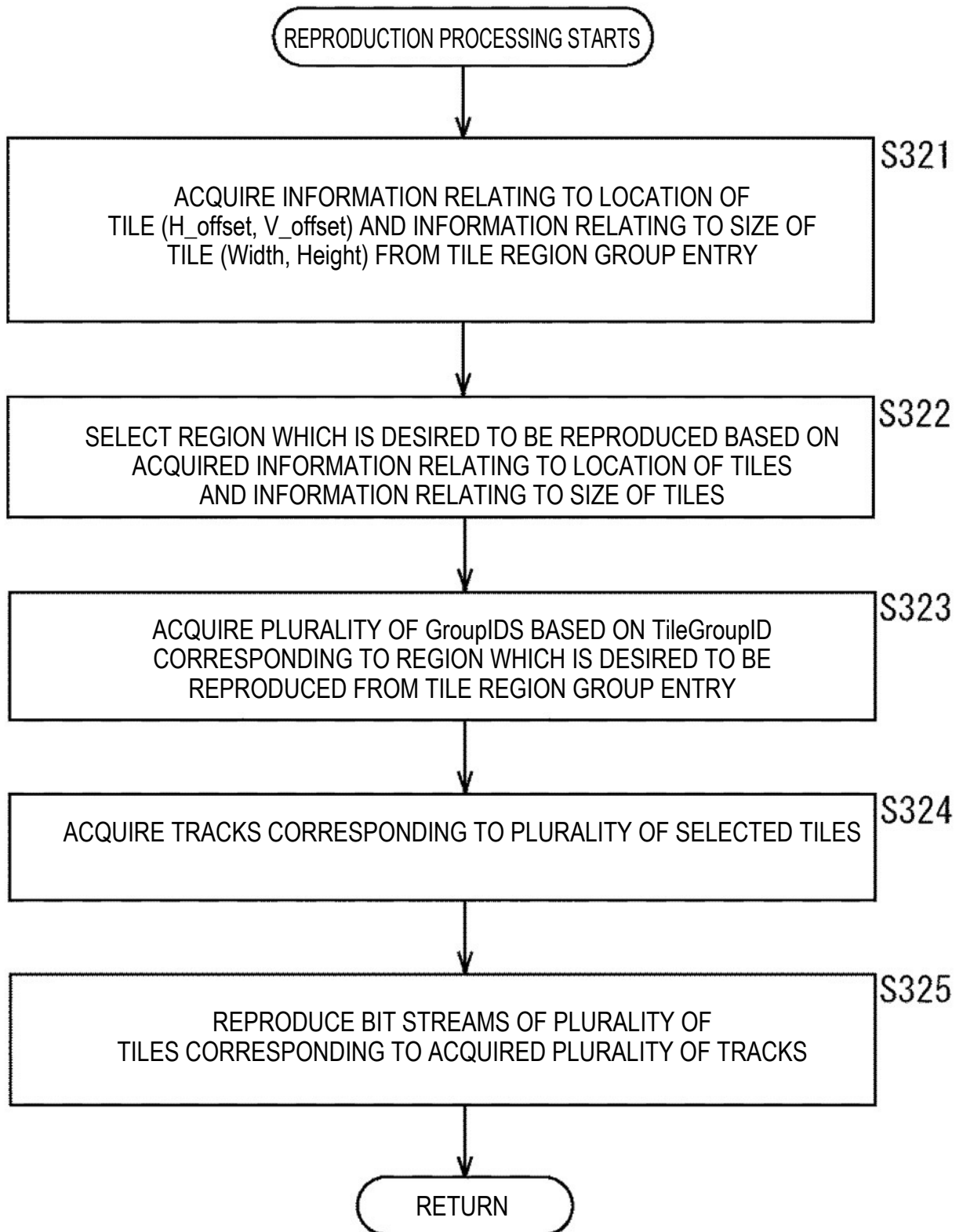
FIG. 48

FIG. 49

