```
Unpack BSI data
For block = 1 to B (the number of blocks)
    Unpack fixed data
    Save pointers to packed exponents
    For chan = 1 to N (the number of coded channels)
        Unpack exponents
        For band = 1 to L (the number of bands)
            Compute bit allocation
            Unpack mantissas
            Unpack coupling channel (save ptrs)
            Scale mantissas / undo coupling
            Denormalize mantissas by exponents
        Compute inverse transform to window domain
        Downmix to appropriate number M of output channel(s)
    For chan = 1 to M (the number of output channels)
        Window & overlap-add with a delay buffer
        Copy downmix buffer values to delay buffer
```

100

FIG. 1
(Prior Art)

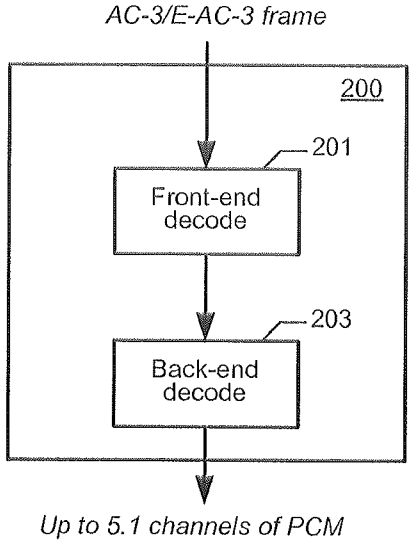

FIG. 2A

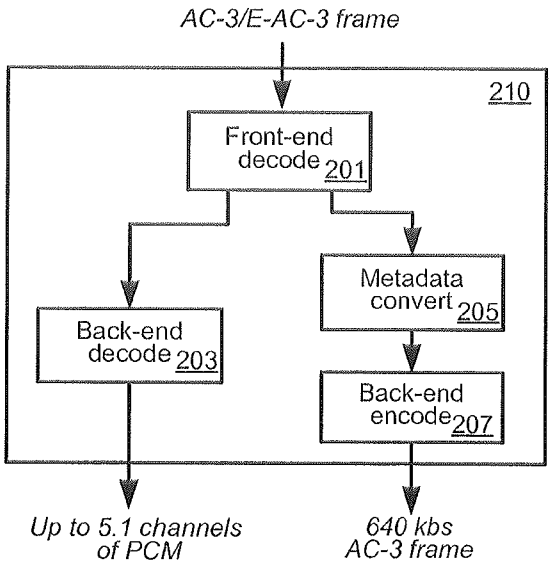

FIG. 2B

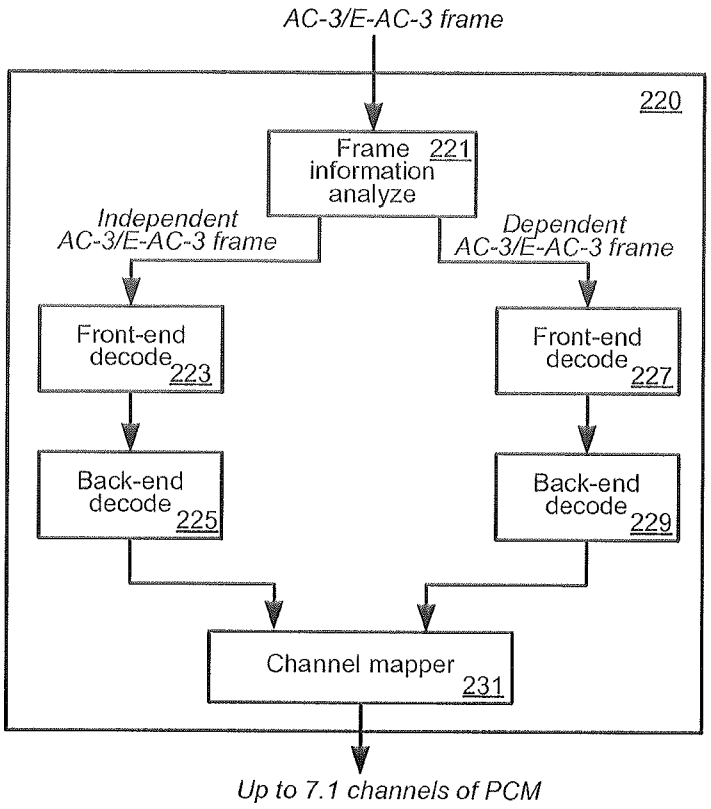

FIG. 2C

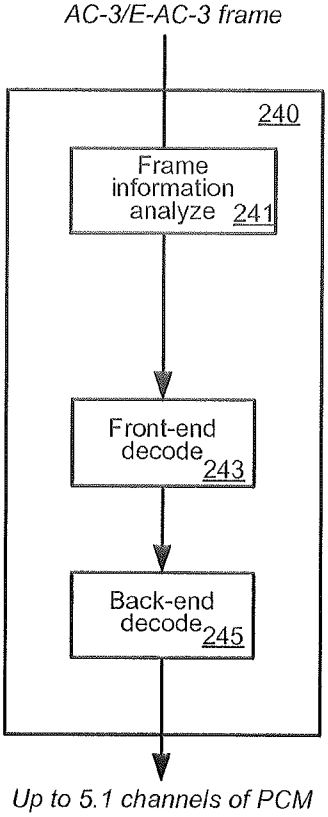

FIG. 2D

Front-end decode 300

```
/* First pass front-end decode */
For block = 0 to B-1 (B=number of blocks)
{
   Unpack fixed data
   For chan = 0 to N-1 (N=number of coded channels)
   {
      Save bitstream pointer to packed exponents
      Unpack exponents
      Save bitstream pointer to packed mantissas
      Compute bit allocation
      Skip Mantissas based on bit allocation
   }
}

/* Second pass front-end decode */
For channel = 0 to N-1 (N=number of coded channels)
{
   For block = 0 to B-1 (B=number of blocks)
   {
      /* unpack */
      Load saved bitstream pointer to packed exponents
      Unpack exponents
      Compute bit allocation
      Load saved bitstream pointer to packed mantissas
      Unpack mantissas
      /* decode */
      Perform standard/enhanced (amplitude-only) decoupling
      Generate spectral extension band
   }
   Transfer exponent and mantissa data from internal to external memory
}
```

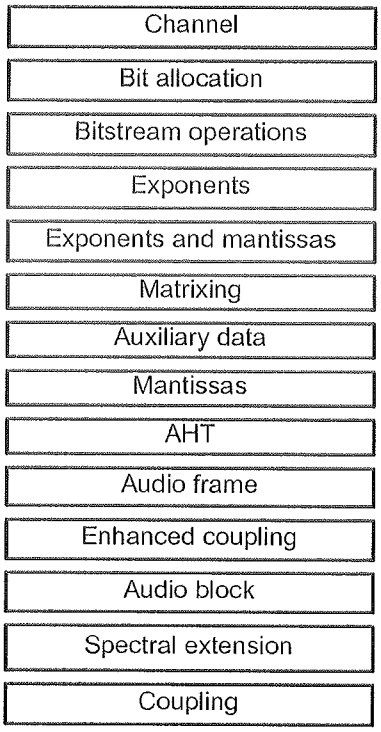

FIG. 3

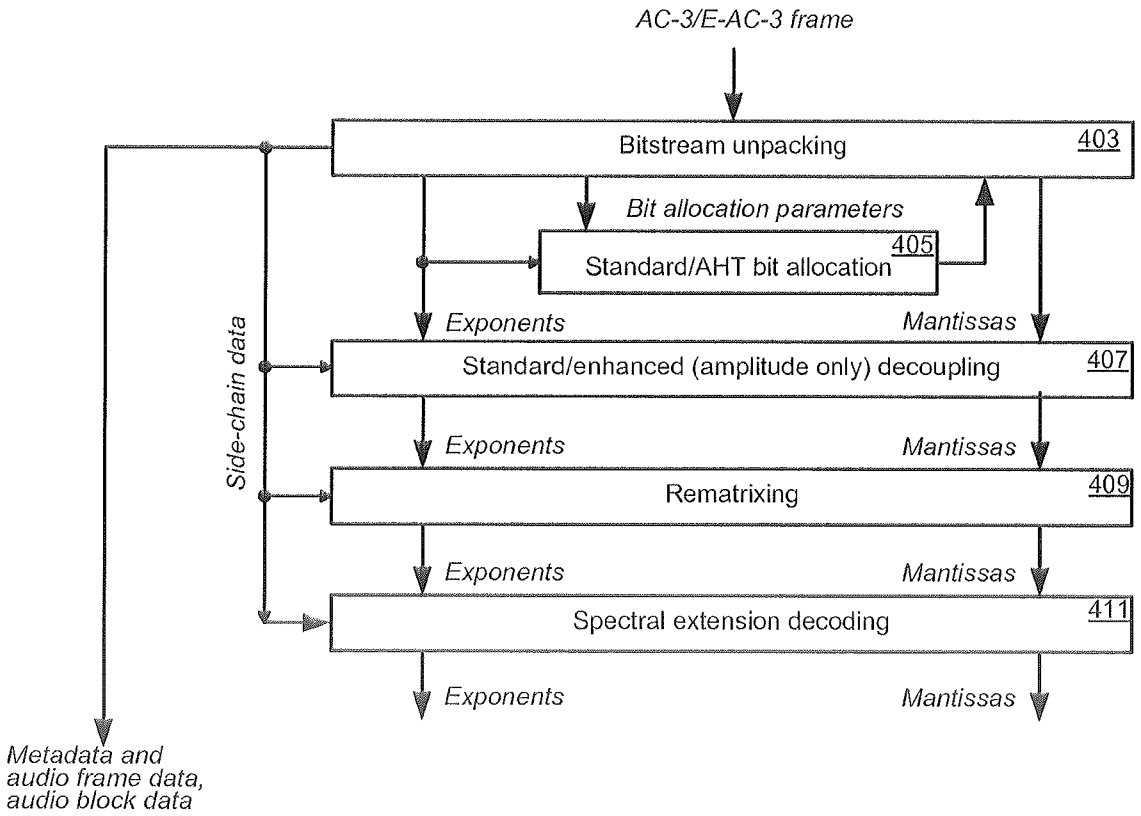
AC-3/E-AC-3 frame
Bitstream unpacking
403
Bit allocation parameters
Standard/AHT bit allocation
405
Exponents
Mantissas
Side-chain data
Standard/enhanced (amplitude only) decoupling
407
Exponents
Mantissas
Rematrixing
409
Exponents
Mantissas
Spectral extension decoding
411
Exponents
Mantissas
Metadata and audio frame data, audio block data

FIG. 4

500

Back-end decode:

```
For block = 0 to B-1 (B=blocks per frame)
{
  Transfer in all blocks of a channel from external memory
  For channel = 0 to N-1 and LFE if n=1 (N.n= number of coded channels)
    {
      Apply dynamic range control, dialog normalization, and gain ranging
      Denormalize mantissas by exponents
      Compute inverse transform
      Window / overlap-add with delay buffer
      Perform transient pre-noise processing
      Downmix to apprpriate output channel(s)
      }
}
```

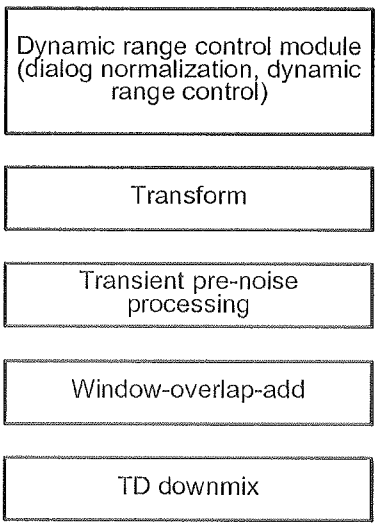

FIG. 5A

520

```
% Back-end decode:

For block = 0 to B-1 (B=blocks per frame)
{
  Transfer in all blocks of a channel from external memory
  Ascertain if FD downmixing or TD downmixing
  If FD downmixing:
   {
     For channel = 0 to N-1 and LFE if n=1
      {
       Apply dynamic range control, dialog normalization, but disable gain ranging
       Denormalize mantissas by exponents
       FD downmix
       Process differently transition block following TD downmixed block
       Deal with any fading out channels
      }
   }
  Else % TD downmixing
   {
     For channel = 0 to N-1 and LFE if n=1  (BUT only for channels in downmix)
      {
       Process differently transition block following FD downmixed block
       Apply dynamic range control, dialog normalization, but disable gain ranging
       Denormalize mantissas by exponents
      }
   }
  For channel = 0 to N-1 and LFE if n=1  (BUT only for channels in downmix; N=M if FD
                                          downmix)
   {
     Compute inverse transform
     Window / overlap-add with delay buffer
     If TD downmix
        {
         Perform transient pre-noise processing
         Downmix to appropriate output channel(s)
        }
   }
}
```

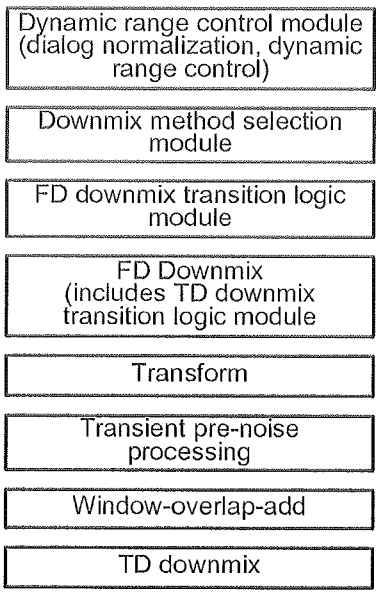

FIG. 5B

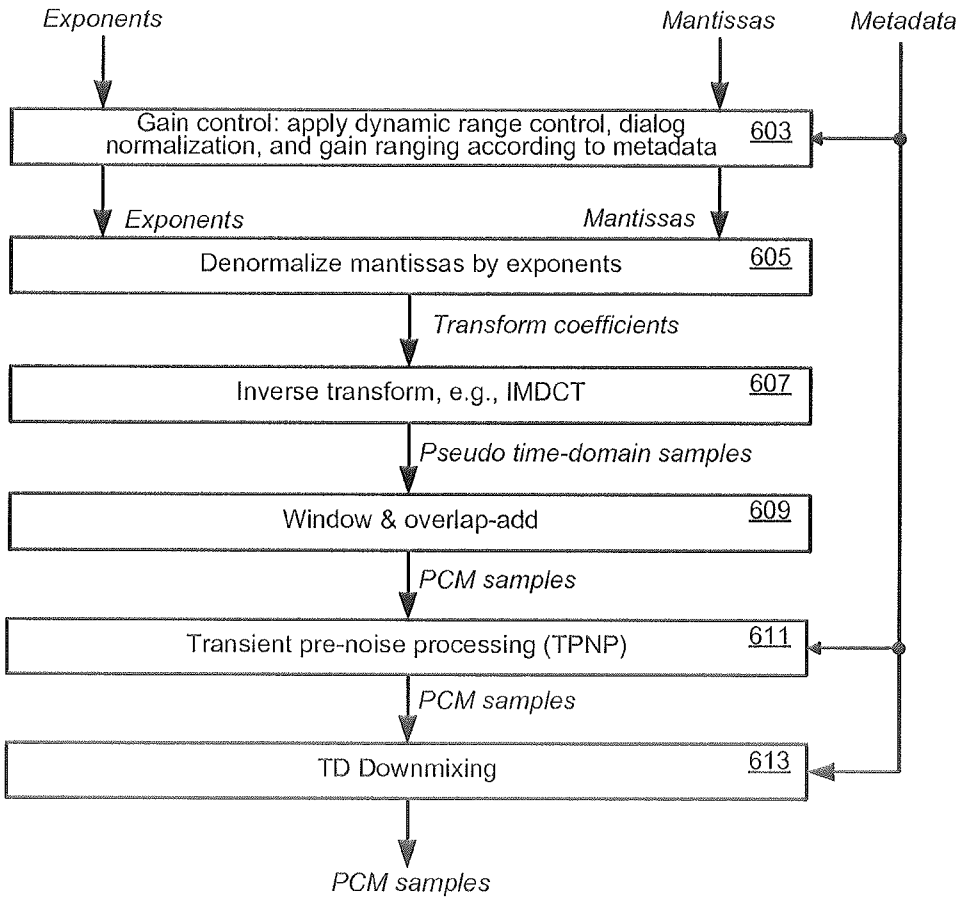
Exponents
Mantissas
Metadata
Gain control: apply dynamic range control, dialog normalization, and gain ranging according to metadata
603
Exponents
Mantissas
Denormalize mantissas by exponents
605
Transform coefficients
Inverse transform, e.g., IMDCT
607
Pseudo time-domain samples
Window & overlap-add
609
PCM samples
Transient pre-noise processing (TPNP)
611
PCM samples
TD Downmixing
613
PCM samples

FIG. 6

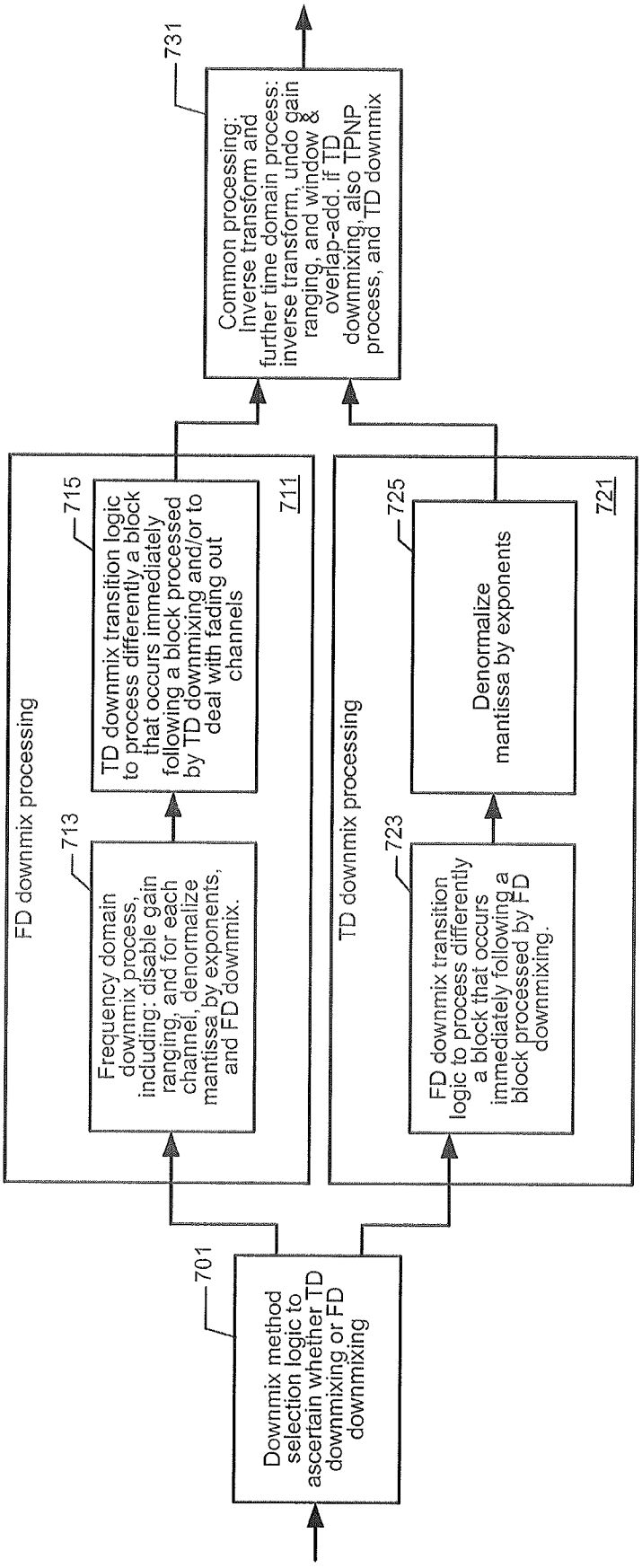
731
Common processing: Inverse transform and further time domain process: inverse transform, undo gain ranging, and window & overlap-add. if TD downmixing, also TPNP process, and TD downmix
715
TD downmix transition logic to process differently a block that occurs immediately following a block processed by TD downmixing and/or to deal with fading out channels
711
FD downmix processing
713
Frequency domain downmix process, including: disable gain ranging, and for each channel, denormalize mantissa by exponents, and FD downmix.
725
Denormalize mantissa by exponents
721
TD downmix processing
723
FD downmix transition logic to process differently a block that occurs immediately following a block processed by FD downmixing.
701
Downmix method selection logic to ascertain whether TD downmixing or FD downmixing

FIG. 7

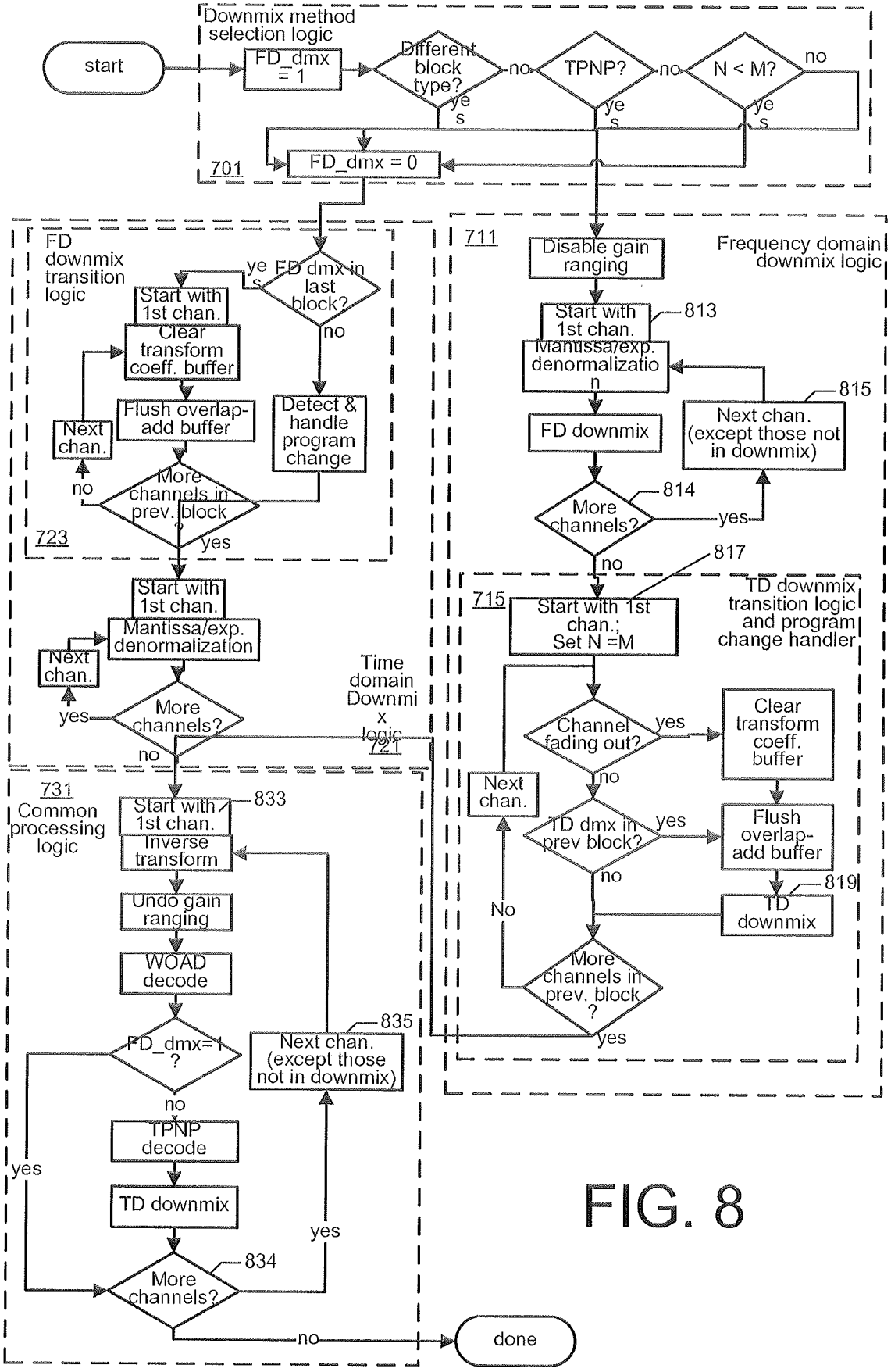
Downmix method selection logic
start
FD_dmx = 1
Different block type?
TPNP?
N < M?
no
yes
FD_dmx = 0
701
FD downmix transition logic
FD dmx in last block?
Start with 1st chan.
Clear transform coeff. buffer
Flush overlap-add buffer
Next chan.
More channels in prev. block?
Detect & handle program change
723
711
Disable gain ranging
Frequency domain downmix logic
Start with 1st chan.
Mantissa/exp. denormalizatio n
813
FD downmix
Next chan. (except those not in downmix)
815
More channels?
814
Start with 1st chan.
Mantissa/exp. denormalization
Next chan.
More channels?
Time domain Downmi x logic
721
817
715
Start with 1st chan.; Set N =M
TD downmix transition logic and program change handler
Channel fading out?
Clear transform coeff. buffer
Next chan.
TD dmx in prev block?
Flush overlap-add buffer
819
TD downmix
No
More channels in prev. block ?
731
Common processing logic
Start with 1st chan.
833
Inverse transform
Undo gain ranging
WOAD decode
FD_dmx=1 ?
Next chan. (except those not in downmix)
835
TPNP decode
TD downmix
More channels?
834
done

FIG. 8

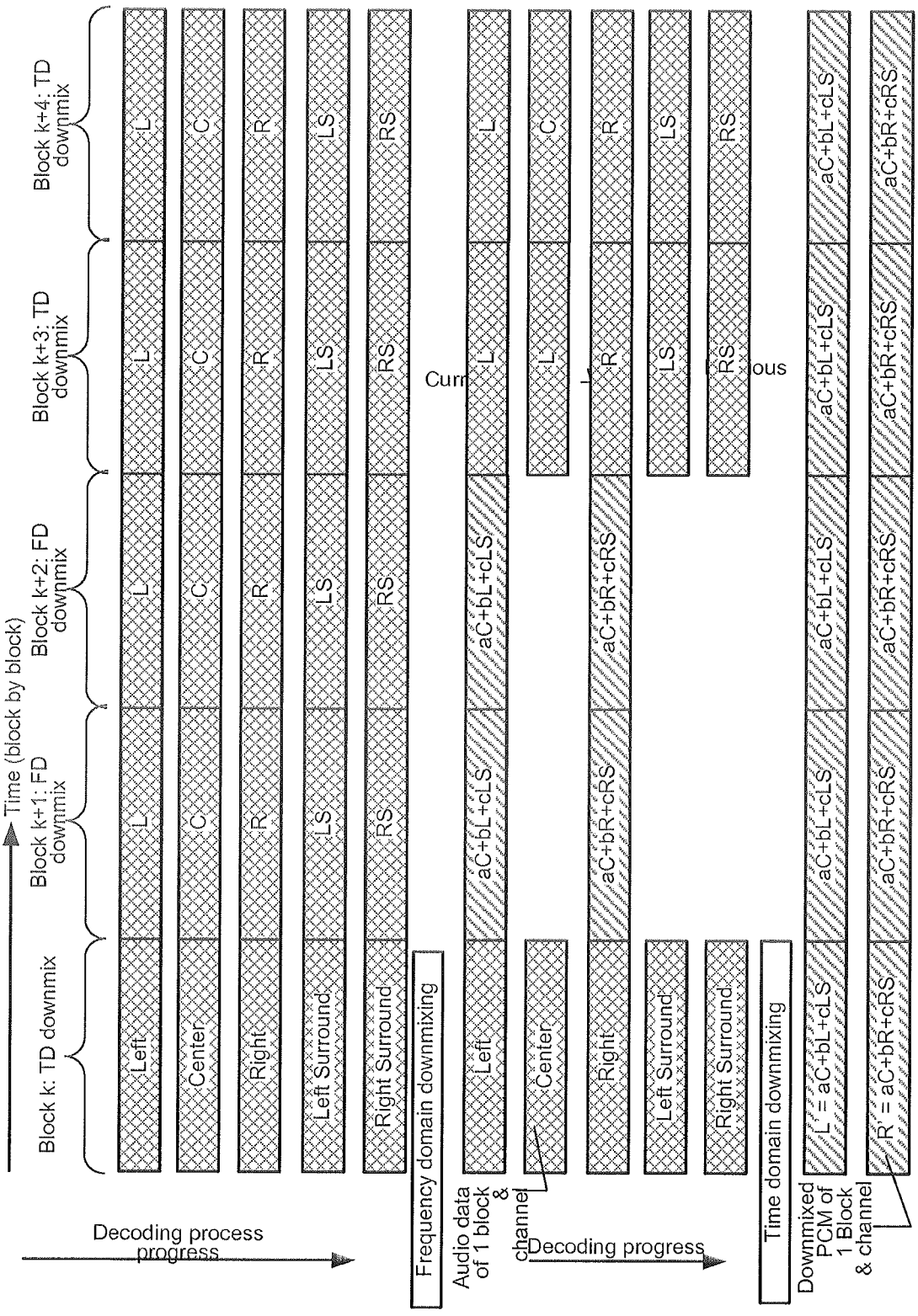
Time (block by block)
Block k: TD downmix
Block k+1: FD downmix
Block k+2: FD downmix
Block k+3: TD downmix
Block k+4: TD downmix
Left
Center
Right
Left Surround
Right Surround
Decoding process progress
Frequency domain downmixing
Audio data of 1 block & channel
Decoding progress
Time domain downmixing
Downmixed PCM of 1 Block & channel

FIG. 9

Time (block by block)
Block k: TD downmix
Block k+1: FD downmix
Block k+2: FD downmix
Block k+3: TD downmix
Block k+4: TD downmix
Left
Center
Right
Left Surround
Right Surround
Decoding process progress
Frequency domain downmixing
Audio data of 1 block & channel
Audio data contribution of the current block k+1
From current block k+3
Audio data contribution of the previous block k (overlap-add buffer)
From previous block k+2
Decoding progress
Time domain downmixing
Downmixed PCM of 1 Block & channel
L' = aC+bL+cLS
R' = aC+bR+cRS

FIG. 10

```
/* TD downmix according to downmixing data*/
if (new downmixing data = old downmixing data)
{
   set up for SSE instructions;
   downmix using downmixing data;
}
else
{
   cross-fade old data to new data using a window
   downmix using cross-faded downmixing data;
}
```

1100

FIG. 11

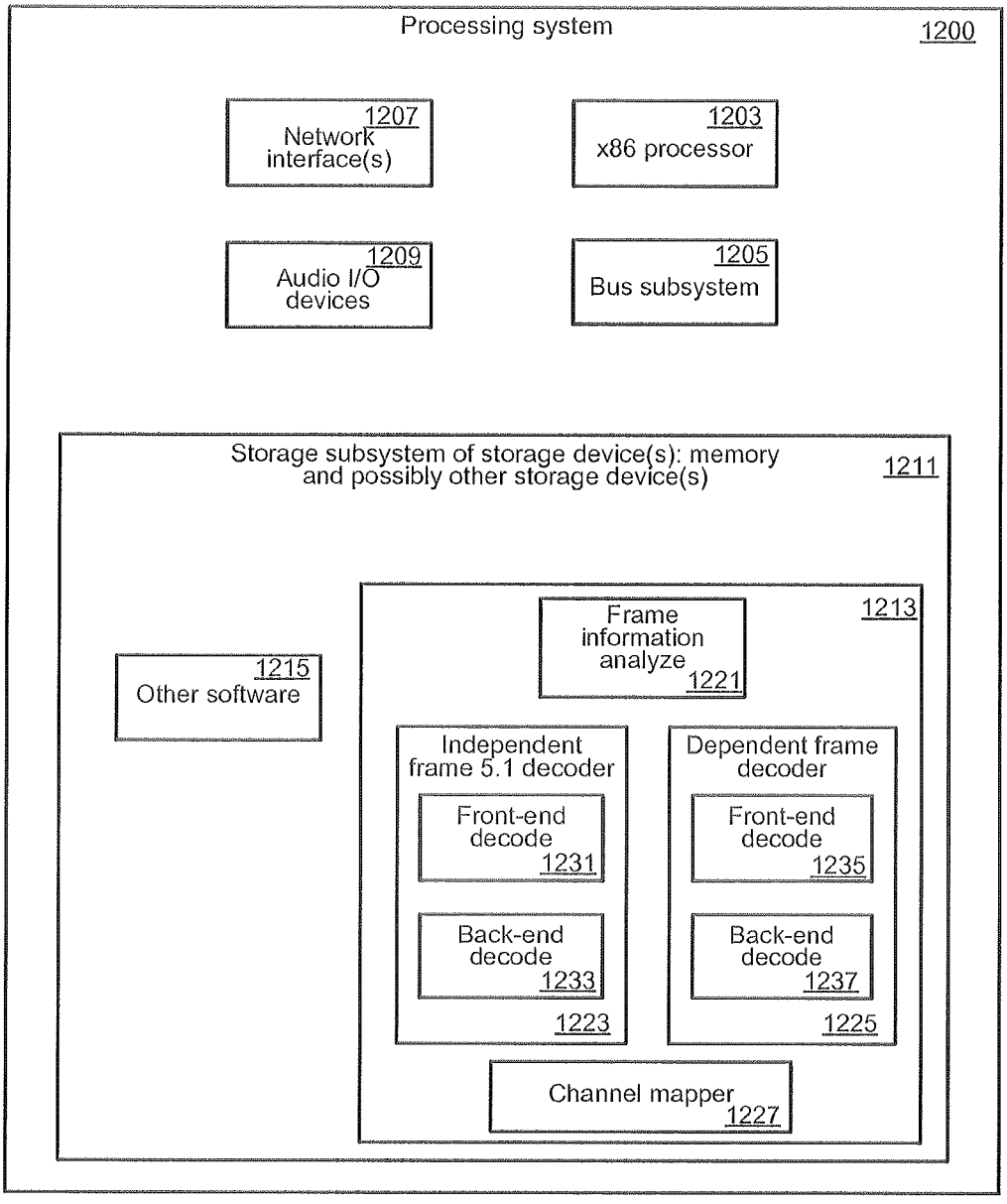
Processing system
1200
1207
Network interface(s)
1203
x86 processor
1209
Audio I/O devices
1205
Bus subsystem
Storage subsystem of storage device(s): memory and possibly other storage device(s)
1211
1213
Frame information analyze
1221
1215
Other software
Independent frame 5.1 decoder
Front-end decode
1231
Back-end decode
1233
1223
Dependent frame decoder
Front-end decode
1235
Back-end decode
1237
1225
Channel mapper
1227

FIG. 12