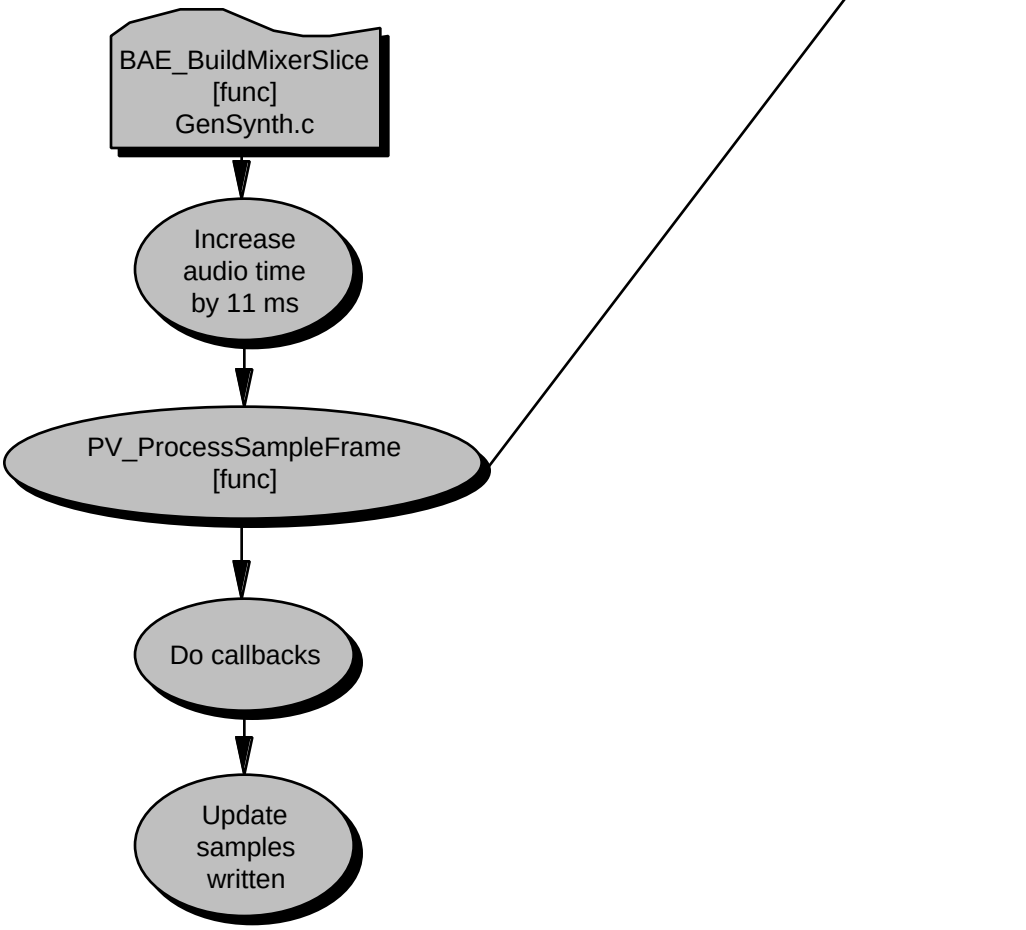
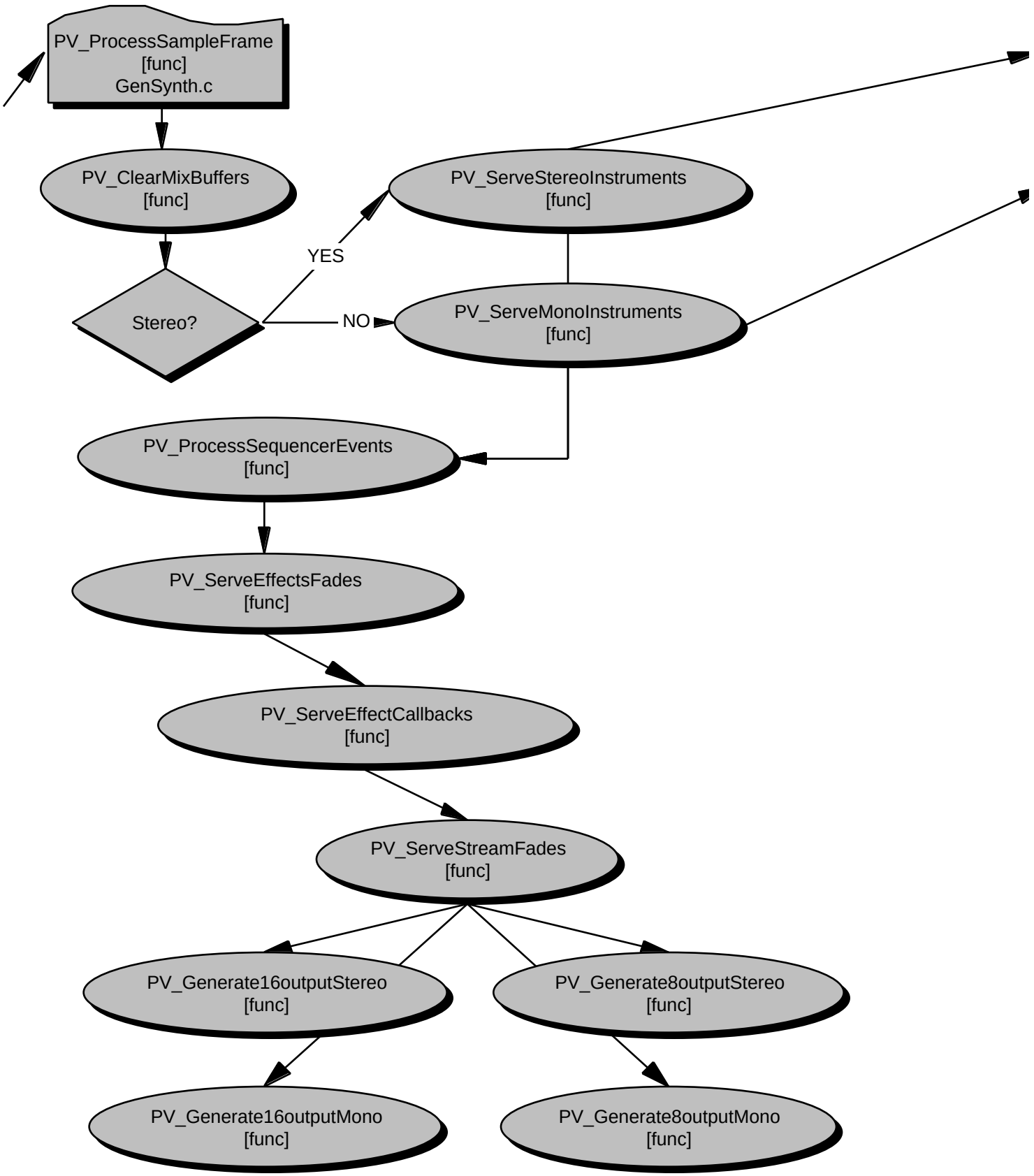
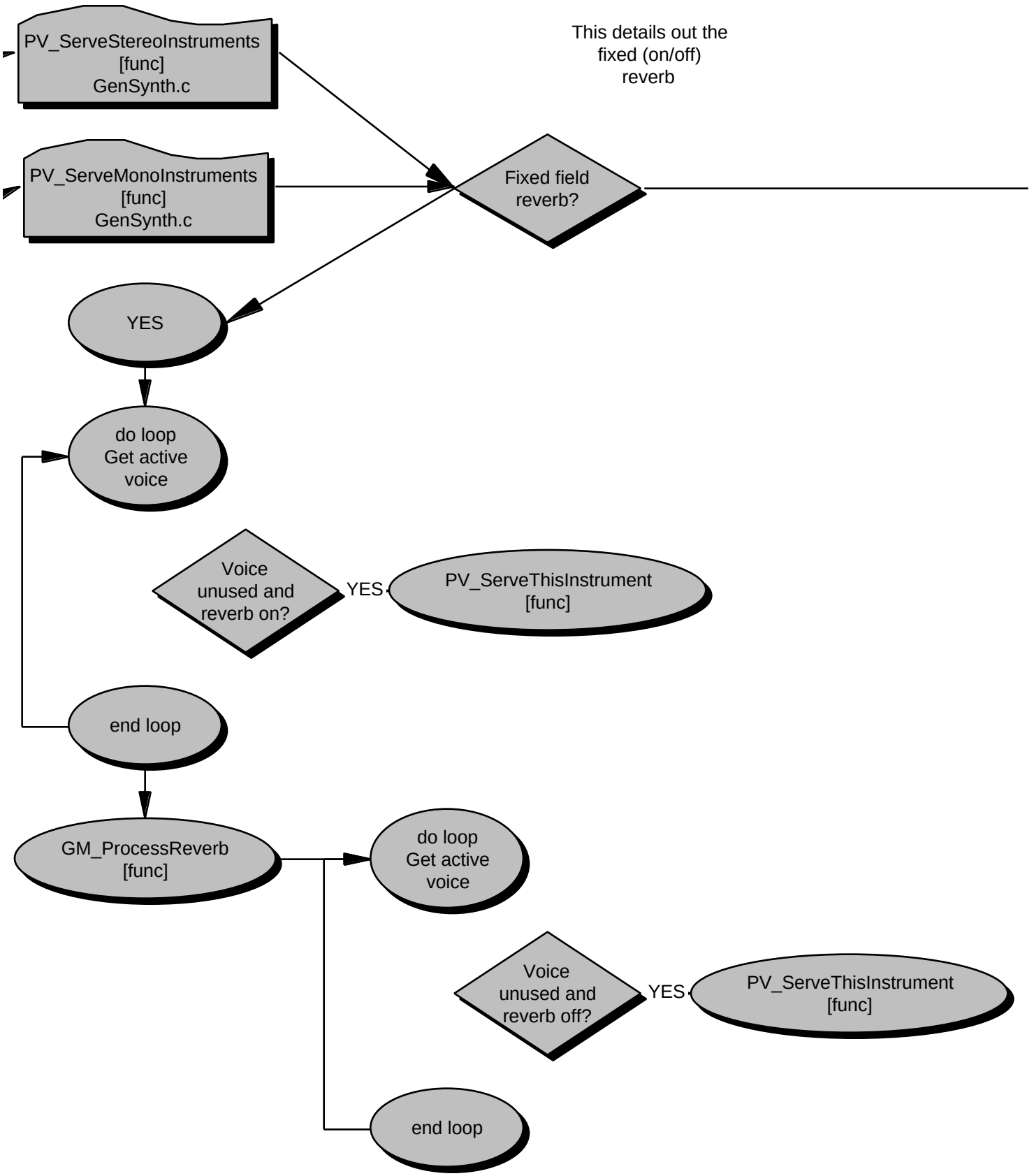
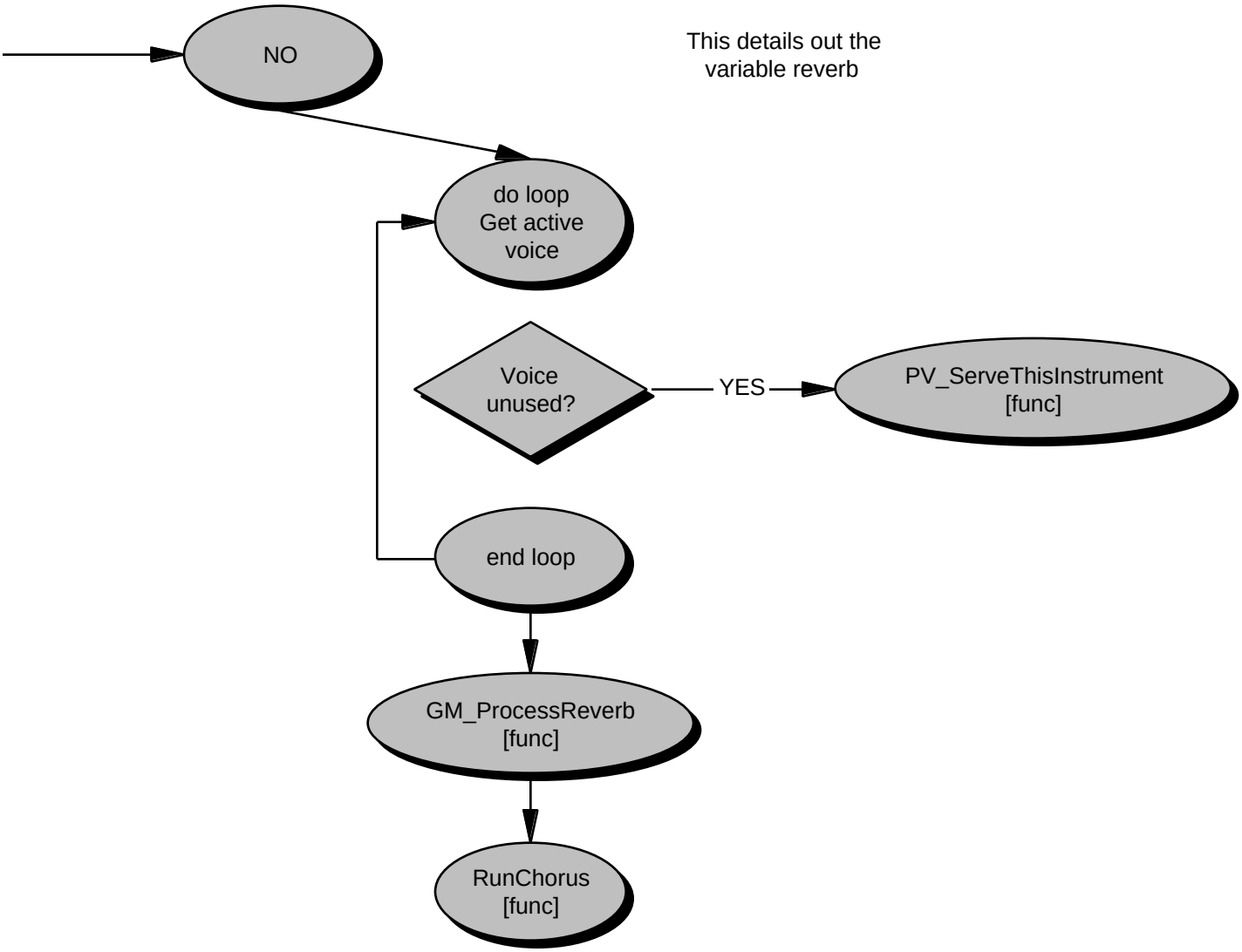
Beatnik Audio Engine

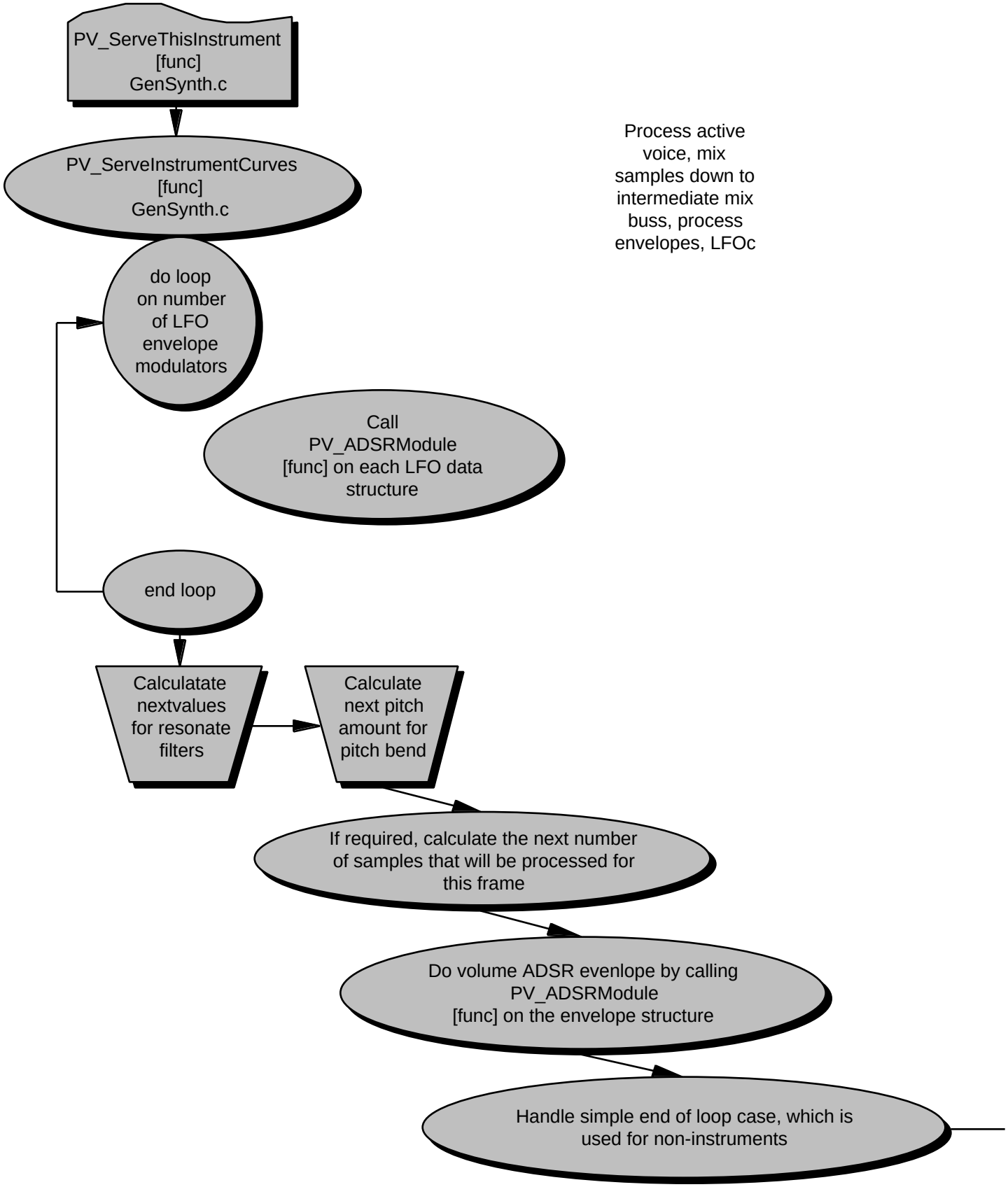
Detail function level starting at
BAE_BuildMixerSlice

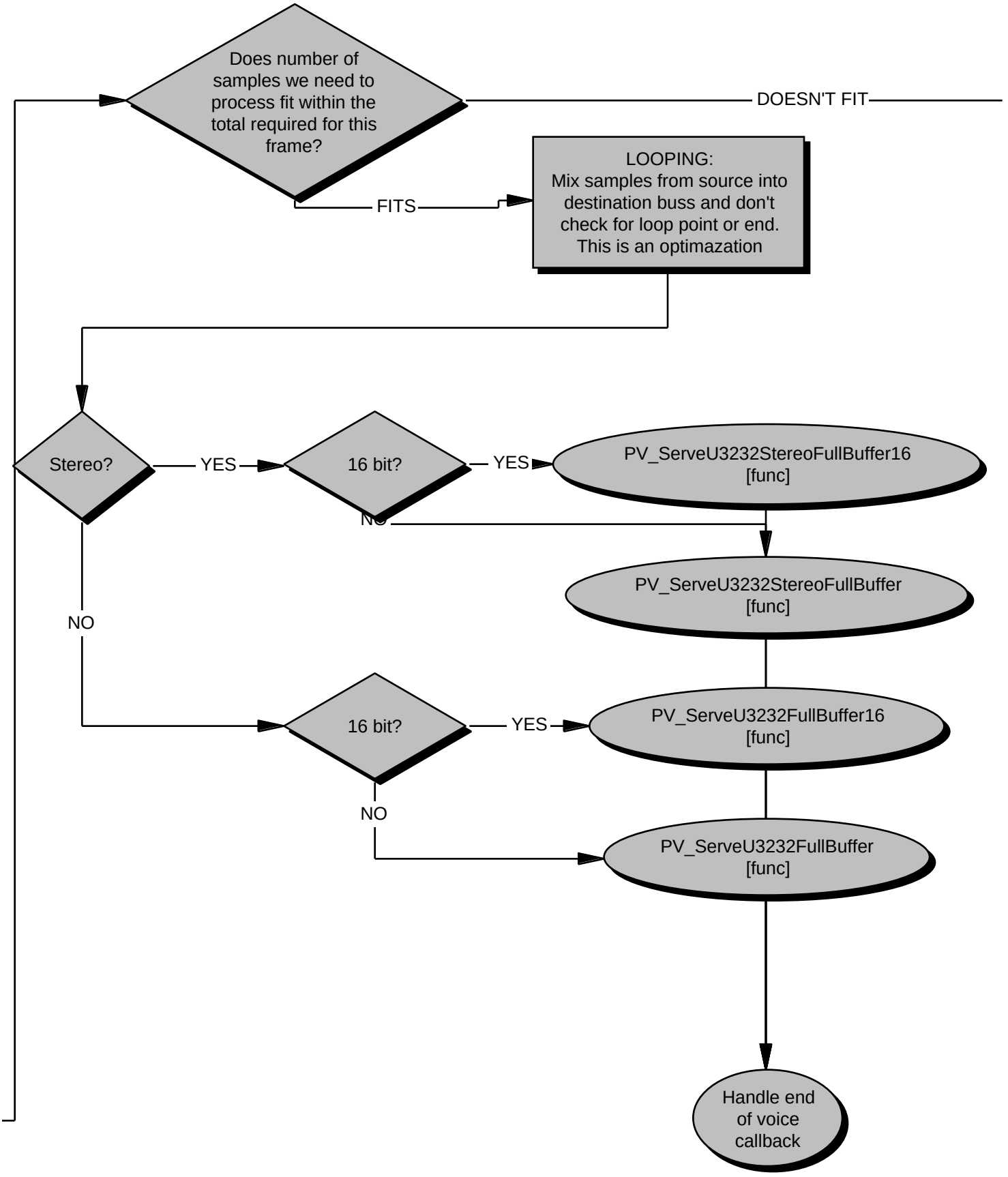


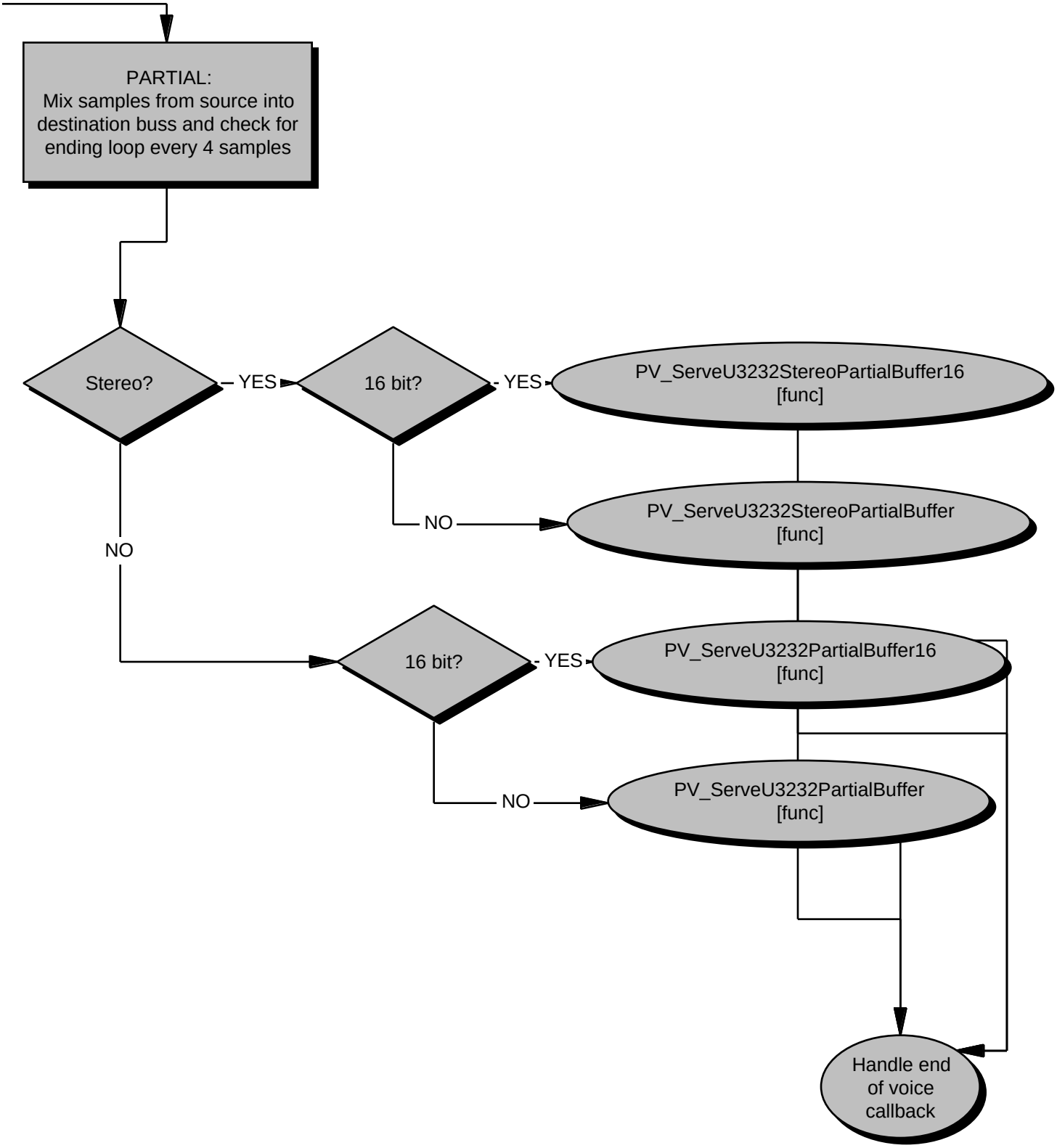

PV_ProcessSampleFrame
[func]
GenSynth.c
PV_ClearMixBuffers
[func]
Stereo?
YES
NO
PV_ServeStereoInstruments
[func]
PV_ServeMonoInstruments
[func]
PV_ProcessSequencerEvents
[func]
PV_ServeEffectsFades
[func]
PV_ServeEffectCallbacks
[func]
PV_ServeStreamFades
[func]
PV_Generate16outputStereo
[func]
PV_Generate8outputStereo
[func]
PV_Generate16outputMono
[func]
PV_Generate8outputMono
[func]


PV_ServeStereoInstruments
[func]
GenSynth.c
PV_ServeMonoInstruments
[func]
GenSynth.c
This details out the
fixed (on/off)
reverb
Fixed field
reverb?
YES
do loop
Get active
voice
Voice
unused and
reverb on?
YES
PV_ServeThisInstrument
[func]
end loop
GM_ProcessReverb
[func]
do loop
Get active
voice
Voice
unused and
reverb off?
YES
PV_ServeThisInstrument
[func]
end loop


NO
This details out the variable reverb
do loop Get active voice
Voice unused?
YES
PV_ServeThisInstrument [func]
end loop
GM_ProcessReverb [func]
RunChorus [func]


PV_ServeThisInstrument
[func]
GenSynth.c
PV_ServeInstrumentCurves
[func]
GenSynth.c
Process active voice, mix samples down to intermediate mix buss, process envelopes, LFOc
do loop on number of LFO envelope modulators
Call PV_ADSRModule [func] on each LFO data structure
end loop
Calculatate nextvalues for resonate filters
Calculate next pitch amount for pitch bend
If required, calculate the next number of samples that will be processed for this frame
Do volume ADSR evenlope by calling PV_ADSRModule [func] on the envelope structure
Handle simple end of loop case, which is used for non-instruments


Does number of samples we need to process fit within the total required for this frame?
DOESN'T FIT
FITS
LOOPING:
Mix samples from source into destination buss and don't check for loop point or end. This is an optimazation
Stereo?
YES
16 bit?
YES
PV_ServeU3232StereoFullBuffer16 [func]
NO
PV_ServeU3232StereoFullBuffer [func]
NO
16 bit?
YES
PV_ServeU3232FullBuffer16 [func]
NO
PV_ServeU3232FullBuffer [func]
Handle end of voice callback


PARTIAL:
Mix samples from source into destination buss and check for ending loop every 4 samples
Stereo?
YES
16 bit?
YES
PV_ServeU3232StereoPartialBuffer16 [func]
NO
PV_ServeU3232StereoPartialBuffer [func]
NO
16 bit?
YES
PV_ServeU3232PartialBuffer16 [func]
NO
PV_ServeU3232PartialBuffer [func]
Handle end of voice callback

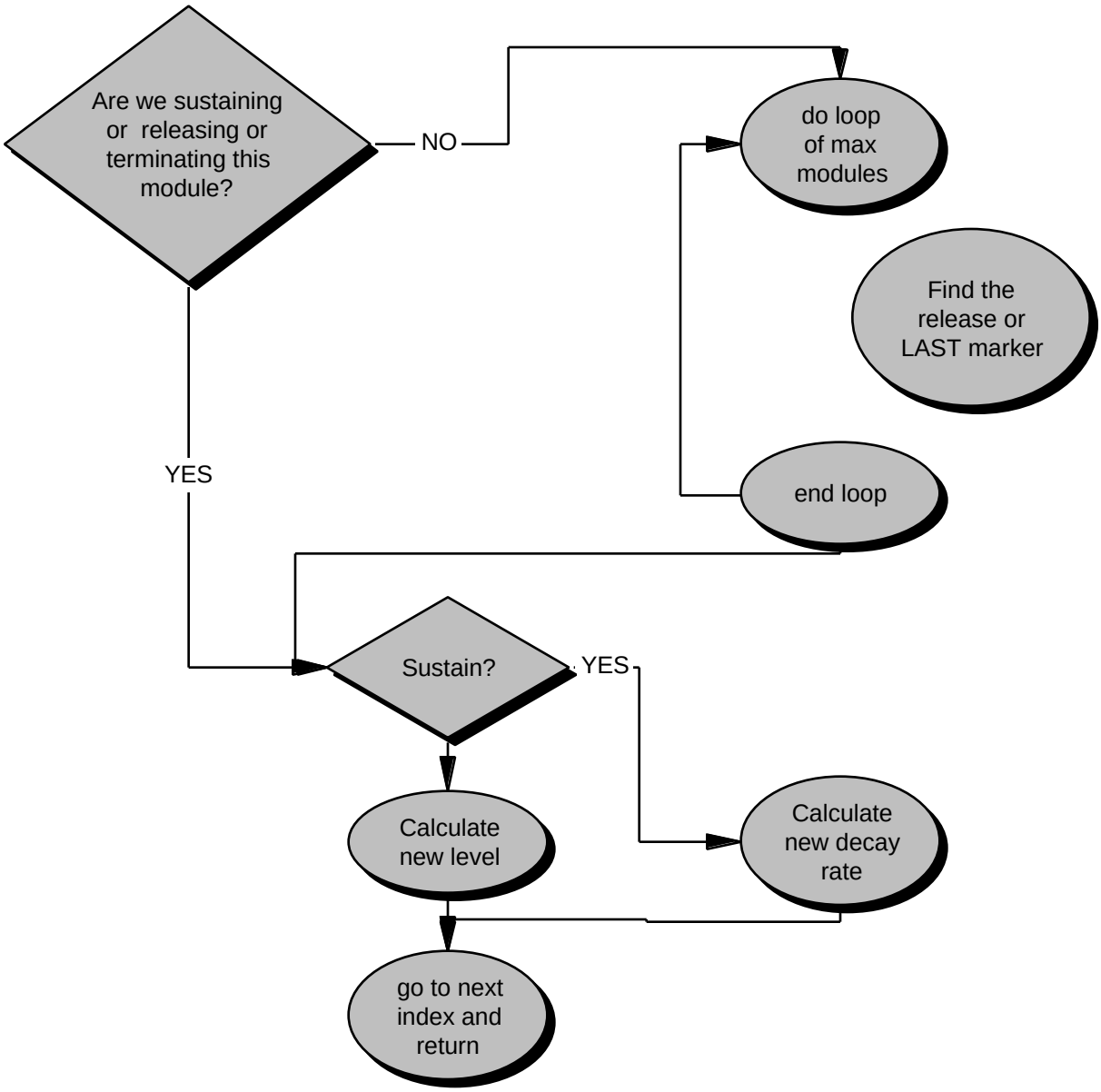
PV_ADSRModule
[func]
GenSynth.c

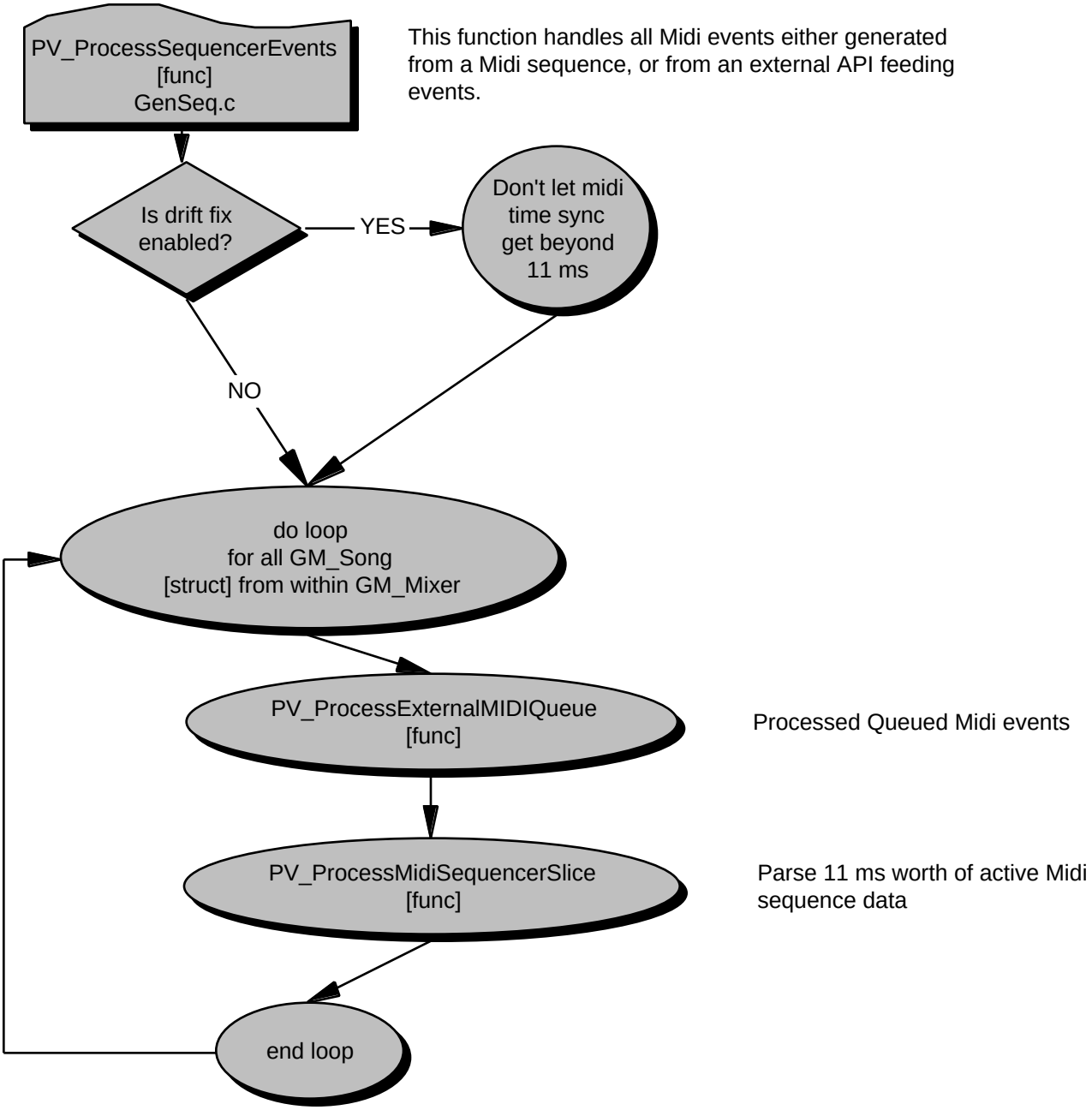
This implements an 8 point envelope with 3 special commands: SUSTAIN, RELEASE, and TERMINATE.

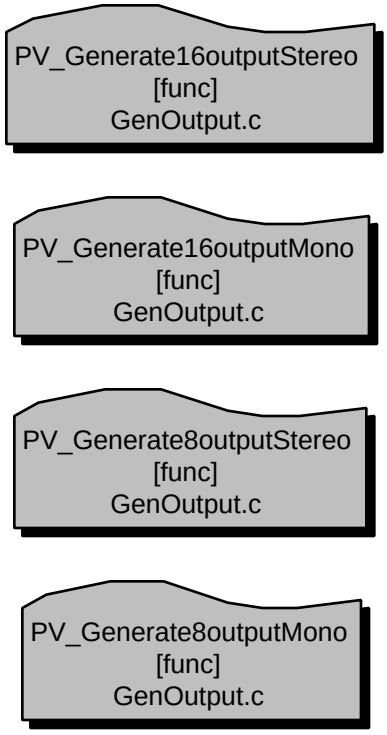
The level point is interpolated between each level point of the envelope.

If a SUSTAIN type is encountered then the state machine loops until it is told to stop, then it jumps to the RELEASE level and continues from there.

Each time the function is called, an index is incremented and a new level is calculated.




PV_ProcessSequencerEvents
[func]
GenSeq.c
This function handles all Midi events either generated from a Midi sequence, or from an external API feeding events.
Is drift fix enabled?
YES
Don't let midi time sync get beyond 11 ms
NO
do loop
for all GM_Song
[struct] from within GM_Mixer
PV_ProcessExternalMIDIQueue
[func]
Processed Queued Midi events
PV_ProcessMidiSequencerSlice
[func]
Parse 11 ms worth of active Midi sequence data
end loop



These 4 functions basicly do the same thing.

They convert from BAE's internal 32 bit mix buss and downsample and down bit size to the final output type required for a particlar sound card

This is one inner loop case.

Specificly output stereo 16 with end loop checking

This example only shows the mono