VS1063A PATCHES

"VLSI Solution Audio Decoder/Encoder"

Project Code: VS1063
Project Name: Support

All information in this document is provided as-is without warranty. Features are subject to change without notice.

Revision History			
Rev.	**Date**	**Author**	**Description**
2.5	2019-12-04	PO	SCI_AUDATA shows 2000Hz after patch started. Versions with spectrum analyzer.
2.4	2019-10-01	PO	ID3v2 parser added. 0x4944 ('ID') in HDAT1 while data is being skipped.
2.3	2019-03-04	POj	Version w/ FLAC decoder with data CRC check.
2.2	2018-08-06	POj	A small fix to MPEG-DASH.
2.1	2018-05-29	POj	SDI FIFO switching fix (e.g. ALAC in M4A).
2.01	2017-03-29	POj	SCI_DECODE_TIME updated during Ogg Vorbis, mp3 and FLAC encoding. FLAC encoder bugfix.
2.0	2017-03-15	POj	A small patch to mp4 decoding to support DASH. Experimental FLAC encoder (format 7) in vs1063a-encpatchesflac.plg . UART-output fixes and additional channel modes enabled for vs1063a-encpatches and vs1063a-encpatchesflac. See section 1.10.
1.9	2015-05-28	POj	Resampler allows rate tune with 48kHz and 12.288MHz XTALI. Also used with short-frame AAC mode.
1.8b	2014-10-30	POj	Higher target level for AGC, better quality for higher maxgain values. Swap channels stereo mode 5.
1.7	2014-07-16	POj	Experimental ALAC decoding from mp4 and CAFF containers. Available in playpatches only.
1.66	2014-04-14	POj	Mono downmix in encoding/codec mode took only the left channel. Added ADC interrupt patch to encpatches.
1.65	2014-03-28	POj	Added version with only play-mode patches. Fixes to WRAM (reading Y and I memory could lock up).
1.6	2013-10-09	POj	WRAM autoincrement didn't work in encoding/code mode. AEC improvements. Faster SCI int. for play mode. Simplified LATM/LOAS format parser (DAB+).
1.5	2013-06-07	POj	Mp3 encoding fix (had noise in high frequencies).
1.4	2012-08-23	POj	AAC: ADTS supports short frame length (parametric_x.config1 bit 9).

Contents

List of Figures

1   Description

There are some known problems in the VS1063a firmware. This patch fixes the issues.

- MP3 Encoder: a wrong Huffman code is generated from one of the tables.
- SetRate: sample-exact samplerate change can go wrong with small probability.
- Encoders: if SCI_VOL is 0, monitoring volume can stay set to mute.
- MP3 Encoder: quantizer of the highest band could get corrupted.
- Codec mode: when not using RIFF WAV header, wrong playback samplerate is used.
- Encoders: if ADC has already been on since last hardware reset, channels can get swapped on ADC activation.
- Codec mode: when not using RIFF WAV header, linear PCM defaults to 0 bits per sample, i.e. no sound produced.
- AAC/MP4: StreamDiscard() missing in header processing.
- FLAC decoding does not set/clear DO_NOT_JUMP during headers.
- InitHardware initializes wrong curFctl/newFctl (2000 Hz).
- Encoders: 44.1kHz rate is not exact, it is actually 44.2kHz.
- Encoders/codec: HDAT0 and HDAT1 interrupt handler takes a lot of cycles.
- ADMixer: configuration from wrong variable, channel-swap
- Max 5.0x clock is possible, encoding combinations may sometimes need more.
- Codec: 16 kHz and 32 kHz rates are not supported.
- During high interrupt loads some interrupts may be missed. This problem mostly occurs with the 44.1 kHz resampler.
- Codec: using AEC crashes the codec mode after a few seconds.
- FLAC: handling of wasted bit does not compensate the level.
- FLAC: Stream buffer management error at FLAC decode startup.
- AAC: ADTS loop did not clear error indicator.
- AAC: ADTS now partly supports short frame length (parametric_x.config1 bit 9).
- AAC/MP4: some atoms were thought to have subatoms when they did not.
- MP3 Encoder: encoding could generate two extra elements, causing noise in the highest frequencies.
- AEC: Improvements. See the AEC application note.
- AAC: A simplified parser of LATM/LOAS format was added.
- AAC: Dynamic Range Control disabled for the time being (patch pending).
- Encoding/Codec: mono-downmix only takes the left channel. (fixed)
- MP4: Minimal patches to add support for DASH.
- Encoding: FLAC encoder. SCI_DECODE_TIME is updated for Ogg Vorbis, mp3, and FLAC encoding.
- ID3v2 can contain data that matches a decodable format. An ID3 parser now skips the extra data correctly.

There are different variants. Normally you would use the vs1063a-patches, which contains everything except ALAC decoding and FLAC encoding.

If you need to support LATM/LOAS format playback, use vs1063a-patches-latm.

If you only use the encoder and don't have enough memory in your controller for the vs1063a-patches version, you can use the stripped-down version vs1063a-encpatches, which only contains fixes for several encoding-related issues. If you want to use FLAC encoding, use vs1063-encpatchesflac. However, these versions do not contain patches for AEC, exact encoding rate option or faster SCI interrupt handler.

If you don't use encoding/codec mode or switch between patches, you can use a the version vs1063a-playpatches, which only includes play-mode patches. This version also includes ALAC decoding from both the mp4 and CAFF containers and the simplified LATM/LOAS container decoding for AAC. (See the previous patches package for a smaller playpatches version.)

If you expect to encounter FLAC files with data errors, you may consider switching to the vs1063a-playflaccrc version when the filename indicates a FLAC file. This patch replaces the FLAC decoder in vs1063a ROM with a version which checks data CRC in addition to data header CRC.

Both old loading tables (`.c`) and new compressed plugin `.plg` are available. The new plugin format is recommended, because it saves data space and future plugins, patches, and application will be using the new format.

Chip	File	IRAM	Description
VS1063A	vs1063a-patches	0x50..ce6	Full patches
VS1063A	vs1063a-patches-latm	0x50..ecb	With LATM/LOAS
VS1063A	vs1063a-encpatches	0x50..4b2	Encoding only
VS1063A	vs1063a-encpatchesflac	0x50..f54	Encoding only, w/ FLAC encoder
VS1063A	vs1063a-playpatches	0x50..eea	Decoding only, w/ ALAC
VS1063A	vs1063a-playflaccrc	0x50..cb4	Decoding only, w/ FLAC CRC
VS1063A	vs1063a-playpatchesspectrum	0x50..ec2	Decoding, ALAC, spectrum
VS1063A	vs1063a-playflaccrcspectrum	0x50..f77	Decoding, FLAC CRC, spectrum

Note that some X and Y data memory is also used.

When the suitable patch is loaded, it starts automatically (writes 0x0050 to AIADDR) and restarts the system (you should wait for DREQ to rise). AIADDR is cleared during start and is not used afterwards.

The patch must be re-loaded after each hardware or software reset. If you replace software reset by writing 0x0050 to AIADDR, you do not need to reload the patch (except after encoding mode, which always needs to end with a standard software reset). Normally software reset is not needed when you use the cancel mechanism to change songs.

To start encoding mode, first set the encoding parameters, then set the SM_ENCODE bit in the SCI_MODE register (without software reset), then write 0x50 to SCI_AIADDR.

Note: after using encoding mode, use a hardware reset or ADMixer may have its channels swapped.

Unless otherwise specified, this patch is **not** compatible with other vs1063a plugins. You need to give a software reset and load the right code when you switch between different plugins.

1.1 Additional Information

If you find a bug, a curious feature, or are unsure how to use VS1063, let us know.

VSDSP Forum is at **http://www.vsdsp-forum.com**/ .

Support e-mail address is **support@vlsi.fi** .

1.2   Fixes in Detail

MP3 Encoder: Audio quality - Wrong Huffman code

A missing bit mask in the Huffman encoding function causes a wrong code to be generated from one of the quantization tables. The problem manifests itself as occasional high-pitched sounds in the playback of the encoded file. The severity and frequency of the problem changes depending on the selected encoding parameters and the audio being encoded.

This patch adds the correct masking operation, restoring the intended audio quality for all audio sources.

SetRate: Incorrect Samplerate

The samplerate nybble in FREQCTLH may sometimes be set wrong, causing incorrect playback rate. **However, we have not been able to trigger the problem in normal slave decoding mode.**

VS1063A provides a sample-exact samplerate change so you can for example mix mp3 files with any samplerate one after another and get the correct-sounding playback. If you turn off resyncs, you can also play short wav files back-to-back without giving cancel between them.

Also volume control is synchronized to output samples to work optimally with bass/treble controls.

Due to missing Disable/Enable in one branch of SetRate function, it can override interrupt routine's change of the FREQCTLH register. The situation occurs when audio buffer is empty when SetRate is called and the clock multiplier does not change. This can happen mainly during the start of mp4 decoding, but we have only been able to trigger the problem in the standalone player/recorder, not in slave decoding mode.

This patch adds the Disable/Enable to the critical section of code.

Workaround without the patch: volume change also updates the FREQCTL value.

Encoders: if SCI_VOL is 0, monitoring volume can stay set to mute.

The encoding mode mutes monitoring through DAC for the first second to give time for the analog circuits and digital filters time to settle before encoding. The normal playback volume set by SCI_VOL should then be restored. Especially with 48 kHz samplerate this does not seem to always happen when SCI_VOL is set to 0.

This patch increases SCI_VOL to 0x0101 in encoding modes, if it sees 0 in the register.

Workaround without the patch: change SCI_VOL after encoding has started.

MP3 Encoder: Quantizer Selector Overwritten

The quantizer selection information of the highest used band could get overwritten, and thus audio between approximately 13 kHz and 16 kHz could get intermittently suppressed, i.e. not encoded.

This patch avoids the overwrite, allowing material containing upto 16 kHz signals to be encoded correctly.

Codec Mode: Wrong Playback Rate when no RIFF Header

The encoding mode sets the DAC to the same rate than the ADC is run at, i.e. 12 kHz, 24 kHz, or 48 kHz.

When codec mode uses RIFF WAV header, the playback rate will be set according to the WAV sample rate field. When RIFF WAV header is not used, the samplerate is not set. This is wrong, the playback should be the same rate as the encoding rate.

This patch fixes the problem and codec mode now sets the playback rate to equal the encoding rate when RIFF WAV header is not in use.

Encoding Mode: Swapped Channels

When ADC has been active and is deactivated then reactivated without a hardware reset in-between, left and right channels can get swapped. If ADMixer is used, deactivated and used later with the same rate, the problem does not appear.

Channels can be swapped in these cases.

- If encoding after using ADMixer without hardware reset

- If encoding after a previous encoding without hardware reset

- If using ADMixer after encoding without hardware reset

- If using ADMixer with two different rates without hardware reset

This patch fixes the problem for the encoding mode. You can start encoding multiple times without a hardware reset in-between. Now this patch also corrects ADMixer.

Codec mode: No Sound for LPCM

When not using RIFF WAV header (headerless mode), the sample size defaults to 0 bits. This causes linear PCM to produce no sound.

With the patch the sample size is 16 bits for linear PCM mode in headerless mode.

AAC/MP4: StreamDiscard() Missing

A missing StreamDiscard() in header processing can with some files cause the "mdat" chunk name to be missing a byte, thus AAC decoding will not start. This patch fixes the problem.

DO_NOT_JUMP not used with FLAC

FLAC decoding does not set/clear DO_NOT_JUMP during headers. So, if you seek in the file during header, the header size fields can be corrupted and header processing does not end.

This patch adds DO_NOT_JUMP handling to FLAC decoding.

InitHardware Initializes to 2 kHz

InitHardware initializes wrong curFctl/newFctl value (2000 Hz), but sets hwSampleRate variable to 8000 Hz. Calling InitHardware twice and then playing 8000 Hz audio results playback with wrong samplerate (2000 Hz instead of 8000 Hz).

The patch fixes the issue, and since version 2.60 you see 2000 Hz in SCI_AUDATA after the patch has started.

Encoders/codec: HDAT0 and HDAT1

Reading 16-bit data through HDAT0 takes about 100 cycles from the interrupt handler, and HDAT1 requires some processing as well. With low data rates this is not a problem, but the 100 cycles of latency is too much for the 44.1 kHz Resampler.

This patch replaces the normal SCI interrupt handler with another version in encoding and codec modes (decoder mode is not changed). The new version handles HDAT0 reads in 45 cycles and HDAT1 reads in 35 cycles. This allows the 44.1 kHz resampler to perform correctly, otherwise it can miss interrupts (DAC, SCI and ADC interrupts have higher priority than SRC).

WRAMADDR and WRAM handling have also been hand-written.

Note: the rewritten SCI interrupt handler is interruptable, so DO NOT in any circumstances perform another SCI operation before DREQ has risen or you have given the appropriate time for the current operation to finish.

ADMixer: Configuration Problem

ADMixer takes channel and samplerate configuration from the wrong variable, thus resulting virtually always in stereo mode and 192 kHz rate. Also, input channels could get swapped when using ADMixer with different samplerates, or using ADMixer after encoding mode without a hardware reset.

This patch reads settings from `adMixerConfig` and also corrects channel swap.

CLOCKF allows max 5.0x clock

You are able to set the internal clock upto $5.0\times$ using the CLOCKF register, but some encoder/codec combinations could use even more.

With this patch the clock adder part is used for both encoding and codec modes. Note this if you use UART TX mode and calculate the UART divider yourself!

Codec mode: using AEC crashes the codec

Using AEC crashes the codec mode after a few seconds for an unknown reason. It seems calling the AEC functions through an extra stub functions prevent the crashing.

Since version 1.60 there are also significant improvements to the AEC. See the VS1063 AEC application note at http://www.vlsi.fi/en/support/applicationnotes.html .

FLAC decoding: wasted bits

The FLAC decoding does not compensate the signal level when "wasted bits" are used. This causes for example the signal level to be halved occasionally.

This patch fixes the problem.

FLAC decoding: buffer management error at decode startup

The FLAC decoding extends the stream buffer from the normal 2048 bytes to 12288 bytes to reduce the required peak data transfer rate. In certain conditions this extend does not work correctly and the FLAC file decoding can get stuck (decoding headers) or causes noise at the start of the playback.

This patch fixes the problem.

AAC: ADTS loop did not clear error indicator

ADTS decoding left error variable un-cleared. If errors were encountered in decoding, for example because of bit errors, the error variable gets set. If the frame sizes are correct (the next header is found where expected), the decoder is not reinitialized, leaving a non-zero value in the error variable.

If the error variable is non-zero and the first element in the AAC frame is a non-audio frame, then the element loop is exited without decoding the audio element.

This patch fixes the problem by clearing the error variable before each ADTS frame.

AAC/MP4: Parsing issue

Some MP4 atoms were thought to have subatoms when they did not, causing too much data to be read. This made the file unplayable. The issue is now fixed.

MP3 Encoder: Extra Elements

The MP3 encoder could generate one too many long (4x) vector-quantized value when it only had two frequency elements to use. This triggered a problem and caused noise in one or two of the highest frequency elements. This issue is now fixed.

Codec Mode: u-law and A-law

If in the codec mode the software decimation ratio is 1 (24000Hz or 48000Hz rate requested), and u-law or A-law format is used, samples are handled for the playback only once per each input sample. Because encoding still takes a bit of time, the playback will starve eventually. This problem is now fixed by using the decimation ratio 2 for u-law and A-law.

Encoding: AGC Target Level

For some signals the Automatic Gain Control (AGC) produces a result that is quite far away from the maximum amplitude. This patch increases the target level and slows down the adaptation, so that the result will have more amplitude.

The default target level control value is 8192. This value is automatically selected when the encoding/codec mode is started, but can be changed by the user from parametric_x.encoding.reserved[0] field at 0x1e2e in X memory. (Write 0x1e2e to SCI_WRAMADDR, then the new value to SCI_WRAM.)

VS1063A Patches   Support

1.3 New and Extended Features

Encoders: 44.1kHz Rate Not Exact

The analog to digital converter (ADC) allows only a few fixed rates, and the audio path resampler allows only integer dividers, so only some sample rates are exact in the encoding and codec modes. For example when you request 24000 Hz, you get the exact rate, but when you request 44100 Hz, the resulting rate is actually 44201 Hz.

This patch adds a hardware/software Resampler, which provides an exact requested rate. The resampler can be disabled by setting AICTRL3 bit 9 (0x0200) during the encoding mode activation. The resampler takes about 15 MHz at 44.1 kHz samplerate. If you use Ogg Vorbis files with 32 kHz or higher sampling rate you probably need $5.5\times$ clock (see $5.0\times$ fix later in this document).

Also, if encoding rate is already exact, the resampler is not activated. SCI_DECODE_TIME is set to 0x8000 when the resampler has been activated.

When you use the resampler, use vs1063a with at least $4.5\times$ clock. Recommended clock setting will be given in vs1063 datasheet when we have verified them.

Codec: 16 kHz and 32 kHz rates are not supported.

Due to the encoder mode always selecting the next highest ADC rate, 16 kHz and 32 kHz are not possible in codec mode.

This patch changes the ADC rate and decimation rate to support also 16 kHz and 32 kHz encoding rates in codec mode.

New ADC modes for codec/encoding modes

ADC channel modes 6 and 7 produce stereo files but select only left (6) or right (7) channel data.

ADC channel mode 5 produces a stereo file (common gain), but swaps the left and right channels.

Application Hook at 0x60

Zero samples are inserted to the audio FIFO when data to be decoded does not arrive quickly enough and the audio buffer gets too empty (less than 64 stereo samples).

However, vs1053 and vs1063 also have DAC FIFO underrun detection, which causes the audio output to behave cleanly even if new samples are not inserted. If there are no new samples to be sent to DAC, the signal will be faded slowly towards zero.

Writing 0x60 to SCI_AIADDR causes the zero sample insertion to be disabled. Additionally whenever any decoder outputs samples, value of 1 will be written to SCI_AICTRL0. The user must clear the value.

Write 0 to SCI_AIADDR to disable these features.

POj

1.4   15/16 Resampler

If you use 12.288 MHz XTALI, the maximum possible samplerate is 48000 Hz. If you play audio from a real-time source, even when the source is an identical device, its rate can be a little lower or higher than our playback rate, due to differencies in crystals and ambient temperatures. If the source rate is higher than we can play, after a while the input buffer will overrun.

One way to resolve this is to use for example 13 MHz crystal so that you can play faster than 48 kHz. When you need to adjust the playback rate above 48000 Hz with 12.288 MHz crystal, you can use the 15/16 Resampler. This converts the audio internally to 45000 Hz, so you have plenty of room to increase the playback rate.

parametric_x.playMode (address X:0x1e09) bit 7 enables the 15/16 Resampler and the appropriate compensation for the samplerate finetuning control. To activate the downsampler, write 0x1e09 to SCI_WRAMADDR, then 0x0080 to SCI_WRAM. The new samplerate tuning becomes active automatically.

To deactivate, write 0x1e09 to SCI_WRAMADDR, then 0x0000 to SCI_WRAM.

The resampler takes about 5 MHz of processing power with 48 kHz samplerate.

parametric_x.playMode (address X:0x1e09) bit 8 enables the Resampler without the samplerate compensation. With this you can test the Resampler. You should hear the pitch go up when you activate the Resampler with this bit. To activate the downsampler without the rate compensation, write 0x1e09 to SCI_WRAMADDR, then 0x0100 to SCI_WRAM. You can also use this mode with AAC Short Frames (see section 1.5).

parametric_x.playMode

The new Resampler control bits are part of the `playMode` parameter.

playMode		
bits	**Name**	**Usage**
8	**PLAYMODE_RESAMPLER_TEST**	Resampler enable w/o rate compensation
7	**PLAYMODE_RESAMPLER_ON**	Resampler enable with rate compensation
6	PLAYMODE_SPEEDSHIFTER_ON	Speedshifter enable
5	PLAYMODE_EQ5_ON	EQ5 enable
4	PLAYMODE_PCMMIXER_ON	PCM Mixer enable
3	PLAYMODE_ADMIXER_ON	AD Mixer enable
2	PLAYMODE_VUMETER_ON	VU Meter enable
1	PLAYMODE_PAUSE_ON	Pause enable
0	PLAYMODE_MONO_OUTPUT	Mono output select

1.5   Support for AAC Short Frames in ADTS (DAB+)

AAC decoding specifies two transform lengths: 1024 sample and 960 sample frames. The former is normally used in all AAC files. The latter short frame length is used in broadcast applications that require audio frames to be an integer multiple of 20ms. DAB+ is one such system.

To play the AAC files using short frame length at the correct pitch, use 13.1072MHz or higher crystal, but report the XTAL in SCI_CLOCKF 6.25% slower (i.e. the ratio 960/1024).

Alternatively, if you can spare the 5 MHz the 15/16 Resampler takes, set the PLAY-MODE_RESAMPLER_TEST bit in the `playMode` parameter. This produces the required compensation ratio, and you can use the default 12.288 MHz crystal.

There are a few bugs in the decoding of short frame length files in the VS1063a ROM firmware. These problems are corrected by this patch when using the ADTS format.

The flag to indicate short frames for the encoded data is only available in the mp4 container format. Systems using DAB+ prefer to use the ADTS container, so the short frame length flag needs to be conveyed to the decoder as side information.

When decoding ADTS, the short frame length flag is read by the decoder from parametric_x.config1 bit 9. To set the decoding into short frame length mode, write (after software reset, patch loading and other initialization) WRAMADDR = 0x1e03, WRAM = 0x0210, to set parametric_x.config1 to specify short frames (0x0200) and do not automatically upsample (0x0010).

Note: the short frame length decoding in the mp4 format is not yet patched. If you have mp4 files with the short frame length flag set, please send it to us as an example.

1.6 LATM/LOAS Parser

Optionally you can select a version that contains a simplified LATM/LOAS parser to support AAC-LC and HE-AAC in that container. Note that the parser is simplified and assumes only a single subframe, program and layer. Other configurations are considered as non-decodeable.

The data should be byte-aligned for the initial detection of the LATM/LOAS format (byte 0x56 followed by a value of 0xe0 through 0xff, which produces the 11-bit sync mark 0x2b7 in the top bits). You should also send at least two zero byte to SDI before the LATM/LOAS data to guarantee that the format is detected immediately from the first frame. If the data is not byte-aligned in this way, send 0x00 0x00 0x56 0xe0 to trigger the format detection. The decoder will then synchronize to the bitstream even if it is not byte-aligned.

SCI_HDAT1 contains 0x4c41 ('LA') when LATM/LOAS decoding is active. Decoding can be ended by the SM_CANCEL bit of the SCI_MODE register.

If no audio configuration is found, synchronization will time-out regardless of parametric_x.resync, and the decoder exits to the main decode loop and SCI_HDAT1 becomes 0. Once playback has started, the normal operation of parametric_x.resync is restored.

When changing streams, it is recommended to exit the LATM/LOAS decoding mode using SM_CANCEL and sending zero bytes until SCI_HDAT1 becomes 0.

If you come across a file that does not play correctly, please contact our support.

Note: the short frame length decoding is patched for the LATM/LOAS format, but not tested. You also need to have a higher XTALI and set CLOCKF correctly for the short frame length files (or use the 15/16 Resampler) to play at the correct pitch. If you have LATM/LOAS files that have the short frame length flag set, please send some of them to us as examples.

1.7 ALAC Decoder for MP4 and CAFF

This patch version adds Apple Lossless Audio Codec format to the list of supported decoding formats. The ALAC audio can be both in a CAFF container, or in an MP4 container.

SCI_HDAT1 contains "aL" (0x614c) for ALAC in a CAFF container (.caf), and "al" (0x616c) for ALAC in an MP4 container (you will see "M4" 0x4d34 until the actual decoding starts). Like with FLAC, you should send 12288 endfillbytes (if needed) for the cancel procedure.

You can not have EarSpeaker nor SpeedShifter active with ALAC decoding. These are turned off automatically when ALAC decoding is started, and restored after decoding of ALAC finishes.

Please note that the support is experimental, so if you have any issues with it, please let us know.

ALAC needs high clock and high data transfer speed to be decoded in real time.

1.8 DASH Parser

Dynamic Adaptive Streaming over HTTP (DASH), is an adaptive bitrate streaming technique. In DASH the content is split into fragments that use different bitrates depending on the speed and contention of the streaming connection.

DASH is not fully compatible with the normal MP4 Audio file structure.

This patch contains the minimal workarounds needed to make vs1063a to play audio from the MPEG4-DASH container. SCI_HDAT1 becomes "DA" (0x4441) if DASH is detected in a mp4 container.

1.9 ID3v2 Parser

An ID3v2 tag can contain text data, but it can also contain image data or some other binary data. When the format identification routine sees data that matches one of the decodable audio formats, it calls the respective decoder. An inadvertent match may happen when the ID3 header contains binary data. If the respective decoder consumes too much data when trying to figure out how to play the data before giving up, the start of the actual audio data also gets eaten, and the file may not play at all.

The added ID3v2 parser detects the ID3 header, then skips the correct number of bytes, so that the format identification routine can start its work from the beginning of the actual audio data.

When ID3 header is detected, the value of HDAT1 changes to 0x4944 ('ID'). You can break out from the data skip loop with SM_CANCEL, so the parser is compatible with the normal end-of-song processing.

The ID3v2 parser is currently enabled in all player patches.

1.10   FLAC Encoder

`vs1063a-encpatchesflac` contains a lossless FLAC encoder. The encoding mode to use the FLAC encoder is 7.

```
#define ENCMODE_FLAC 7
```

All the normal encoding mode features are available: channel mode selection, including the new channel modes, selectable samplerate and UART output for the encoded data.

Note that FLAC encoding is lossless, so the bitrate can be as high as straight-up uncompressed PCM. Stereo 48 kHz 16-bit file can thus be 1.536 Mbit/sec.

The value in the low 3 bits of quality setting (SCI_WRAMADDR) affects the encoding like the following:
0 = always mid/side - fastest
1 = reserved
2 = chooses between mid/side and discrete channels
3 = reserved
4 = chooses between all 4 channel modes - slowest
5-7 = reserved.

Let us know of your experiences with the FLAC encoder, good or bad.

1.11   FLAC Decoder with CRC

Due to code space reasons the FLAC decoder in vs1063a ROM doesn't check the CRC of the data portion of the FLAC audio frames, only the CRC of each frame header.

If you expect to encounter data errors regularly, there is one variant which adds data CRC integrity checking. If the CRC does not match, the audio block will be discarded.

Unfortunately you can't have both FLAC w/ CRC and ALAC at the same time. We suggest you check the suffix of the filename, and if .flac or .fla, load the corresponding patch variant before starting to decode the file.

1.12   Spectrum Analyzer

Two versions include the spectrum analyzer plugin. The 15/16 Resampler cannot be used with these versions. Also, the `vs1063a-playpatchesspectrum` does not have the LATM/LOAS container support.

The spectrum analyzer can be started by writing 0xd00 to SCI_AIADDR.

The spectrum data is found at 0x1800 and band data at 0x1868. See the spectrum analyzer plugin document for more details.

2   How to Load a Plugin

A plugin file (.plg) contains a data file that contains one unsigned 16-bit array called plugin. The file is in an interleaved and RLE compressed format. An example of a plugin array is:

```
const unsigned short plugin[10] = { /* Compressed plugin */
  0x0007, 0x0001, 0x8260,
  0x0006, 0x0002, 0x1234, 0x5678,
  0x0006, 0x8004, 0xabcd,
};
```

The vector is decoded as follows:

1. Read register address number `addr` and repeat number `n`.
2. If (`n & 0x8000U`), write the next word `n` times to register `addr`.
3. Else write next `n` words to register `addr`.
4. Continue until array has been exhausted.

The example array first tells to write 0x8260 to register 7. Then write 2 words, 0x1234 and 0x5678, to register 6. Finally, write 0xabcd 4 times to register 6.

Assuming the array is in `plugin[]`, a full decoder in C language is provided below:

```
void WriteVS10xxRegister(unsigned short addr, unsigned short value);

void LoadUserCode(void) {
  int i = 0;

  while (i<sizeof(plugin)/sizeof(plugin[0])) {
    unsigned short addr, n, val;
    addr = plugin[i++];
    n = plugin[i++];
    if (n & 0x8000U) { /* RLE run, replicate n samples */
      n &= 0x7FFF;
      val = plugin[i++];
      while (n--) {
        WriteVS10xxRegister(addr, val);
      }
    } else {            /* Copy run, copy n samples */
      while (n--) {
        val = plugin[i++];
        WriteVS10xxRegister(addr, val);
      }
    }
  }
}
```