VS1011/VS1002 Loudness Extension

"VLSI Solution VS1011/VS1002"

Project Name: VSMPG

Revision History			
Rev.	**Date**	**Author**	**Description**
1.41	2003-09-29	HH	First version for VS1011/VS1002.

Contents

List of Figures

1 General

This document is a specification for the Loudness firmware extension that allows for MP3 playback with a high- and low-frequency enhancer.

1.1 References

1. VS1011 and VS1002 Datasheets, version 0.2 (http://www.vlsi.fi/)

1.2 Definitions

B Byte, 8 bits.

b Bit.

IC Integrated Circuit.

Ki Kibi (1024).

k SI kilo (1000).

Mi MeBi ($1024 \times 1024 = 1048576$)

M SI mega (1000000).

VS_DSP[4] VLSI Solution's 16-bit DSP core that is used in VS1011/VS1002.

VSKIT VLSI Solution's VS_DSP tools package.

W Word. VS_DSP[4] data words are 16 bits wide, and instruction words are 32 bits wide.

1.3 Conventions

As a default, all numbers are in decimal format. Hexadecimal numbers are marked with a leading "0x", e.g. "0x1234" (4660), and binary numbers are marked with a leading "0b", e.g. "0b11001010" (202).

In register descriptions, bit masks are always given in binary format, even if not preceded by a "0b".

32-bit numbers are stored in little-endian fashion. Thus, if a 32-bit number is stored in e.g. registers 10..11, the LSBs are at offset 10, and MSBs are at address 11.

2 Running Custom Code on VS1011 and VS1002

As a default, VS1011/VS1002 have 1.25 KiW (minus 32 W) of free instruction memory for user applications, as well as 256 W of free data memory, evenly divided between X and Y data memory.

SCI registers WRAMADDR and WRAM can be used to load code to the internal instruction RAM memory. SCI registers AICTRL[0...3] are left free for user applications.

By setting SCI register AIADDR the user may activate an application program that acts like a filter when decoding MP3 files (see Chapter 4).

All these features mentioned in the VS1011/VS1002 Datasheets are being used to gain control over the standard audio decoding process.

3 Features and Restrictions

The following features and restrictions apply with the VS1011/VS1002 Loudness extension.

Features:

- Loudness unit may be turned on and off on the fly.
- Code needs only to be loaded once to RAM.
- Requires roughly 1.7 MIPS for a 44.1 kHz stereo signal.
- Requires roughly 380 words (1.5 KiB) of VS1011/VS1002 instruction RAM space.
- May be used in at the same time with VLSI Solution's PCM Extension software package.

Restrictions:

- So as not to cause excessive distortion to sound, no frequency is amplified much. Thus, when Loudness is activated, sound level decreses slightly. This can be compensated by turning volume up a few dB when Loudness is activated. See Chapters 6 and 7 for more details.

4 Signal Paths

4.1 VS1011/VS1002 Signal Path

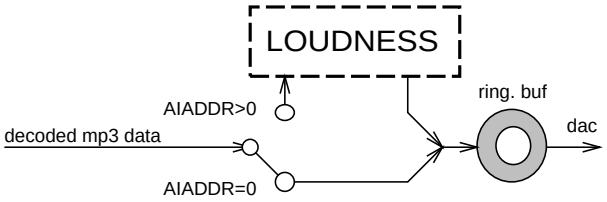

Figure 1: Loudness Signal Path.

After activating Loudness, the decoded MP3 signal is provided to the Loudness software as shown in Figure 1.

4.2 Filters 1...4 Signal Path

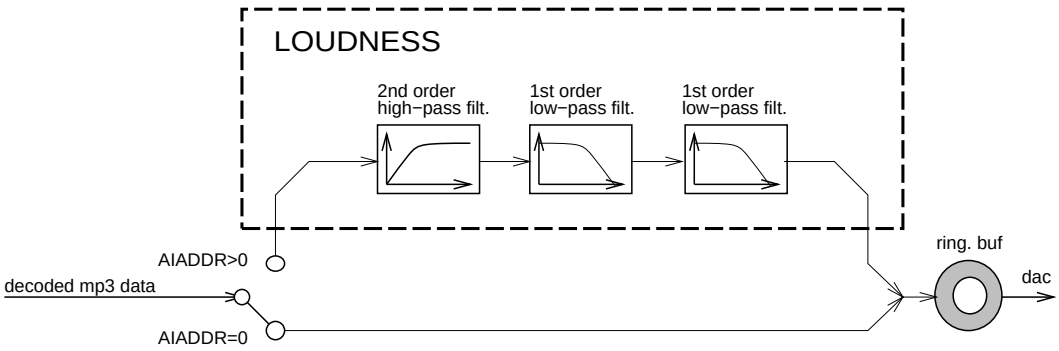

Figure 2: Loudness Structure for Filters 1...4.

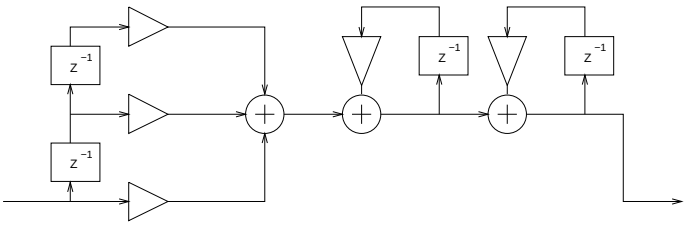

Figure 3: Loudness Implementation for Filters 1...4.

Figure 3 shows the signal path for filters 1...4. The signal is first fed to a 2^{nd} degree FIR filter for high frequencies, and then to two consecutive 1^{st} degree IIR filters to low frequencies.

4.3 Filters 5...12 Signal Path

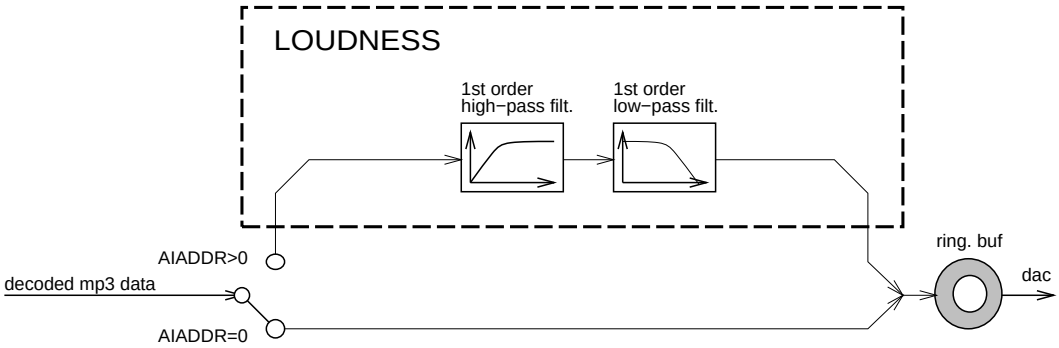

Figure 4: Loudness Structure for Filters 5...12.

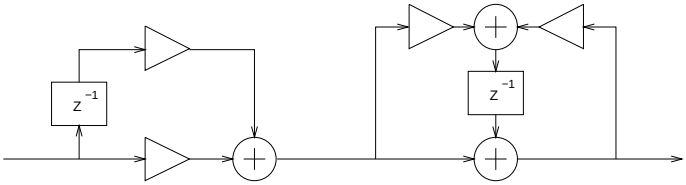

Figure 5: Loudness Implementation for Filters 5...12.

Figure 5 shows the signal path for filters 5...12. The signal is first fed to a 1^{st} degree FIR filter for high frequencies, and then to two consecutive 1^{st} degree IIR filters to low frequencies.

As can be seen, Filters 5...12 use slightly less complex structures for filtering than Filters 1...4. They also have slightly less noise. Thus, if the frequency responses (see Chapter 6) for filters 5...12 seem suitable, use of them is slightly preferred over filters 1...4.

5 User Interface

5.1 VS1011/VS1002 Register Name Redefinitions

Reg. no	Reg. name	Name used in this document
13	AICTRL[0]	LOUD_SET
14	AICTRL[1]	LOUD_UNUSED1
14	AICTRL[2]	LOUD_UNUSED2
14	AICTRL[3]	LOUD_UNUSED3

5.2 LOUD_SET

To reset Loudness, write 0 to this register. After an initial reset, writing a number between 1 and 12 will activate one of the available filters. For filter characteristics, see Chapter 6.

Note: you may switch between filters while you are on the run: you don't have to reset Loudness by writing 0. However, if you have disabled Loudness for a while, it would be best to first reset it with 0 before reactivating to make any possible snaps lower in intensity. See examples in Chapter 7 for details.

Note: If VS1011/VS1002 is fed with a bad MP3 file, it may run a software reset to get back into sync. This will not stop Loudness software from running, but it will reset the filter coefficients. To prevent this from quieting the whole MP3 bitstream, rewrite this register once in a while, for instance once a second or after each successive 10 kB of MP3 data. Write only the filter number value: no 0 initialization is required.

5.3 LOUD_UNUSED[1...3]

Registers are not used by Loudness.

6 Filter Frequency Responses

6.1 Numerical Frequency Responses

There are 12 different filters to choose from. Their characteristics are as follows.

No	Bass/dB	Mid/dB	High/dB	Offset/dB	See
1	-2.5	-16.5	2.5	10.0	Figure 6
2	-1.5	-8.5	2.0	5.0	Figure 7
3	-6.0	-9.5	2.5	5.0	Figure 8
4	3.5	-6.0	2.0	3.0	Figure 9
5	-2.5	-6.0	-6.0	5.0	Figure 10
6	-1.5	-5.0	-2.5	3.0	Figure 12
7	-2.5	-7.0	-3.5	5.0	Figure 14
8	-1.0	0.0	-1.5	0.0	Figure 16
9	0.5	-5.5	-6.0	5.0	Figure 11
10	1.5	-4.5	-2.5	3.0	Figure 13
11	0.0	-7.0	-3.5	5.0	Figure 15
12	-8.0	0.0	-2.5	0.0	Figure 17

No is the filter number (to be written to LOUD_SET). *Bass* is measured at 100 Hz, *Mid* at 600 kHz (Filt 1...4) or 1 kHz (Filt 5...12), *High* at 10 kHz.

Because Loudness mostly attenuates frequencies, VS1011/VS1002 volume has to be increased when activating one of these filters. The field *Offset* shows an estimate of how much volume has to be increased in order to get roughly the same volume as before activating Loudness. Note, that the exact amount that has to be added depends on earphones and a listener's personal preferences, so these values may be adjusted slightly.

Filters 5...12 are paired, in such a way that Filter 9 is a slightly stronger version of filter 5, Filter 10 is a stronger version of Filter 6, etc. This feature can be used for instance by having a two-step "strongness" control, that would control how much of the filter effect in question is required. Example: Filter 8 is a mild band-pass filter, and to get the same effect stronger, switch to its pair, Filter 12.

Loudness filters have been matched for 44.1 kHz operation. However, the frequency responses also work pretty well when a sample rate of 48 kHz or 32 kHz is used. The main difference is that the mid-point frequency varies slightly from the original 1000 Hz to either 700 Hz at 32 kHz, or to 1100 Hz at 48 kHz. This should be hardly noticeable.

Loudness is not intended to be used with sample rates lower than 32 kHz. The audio quality with lower sample rates is going to be inferior anyway. Also, there are very few MP3's around with sample rates other than 44.1 kHz.

6.2 Graphical Frequency Responses

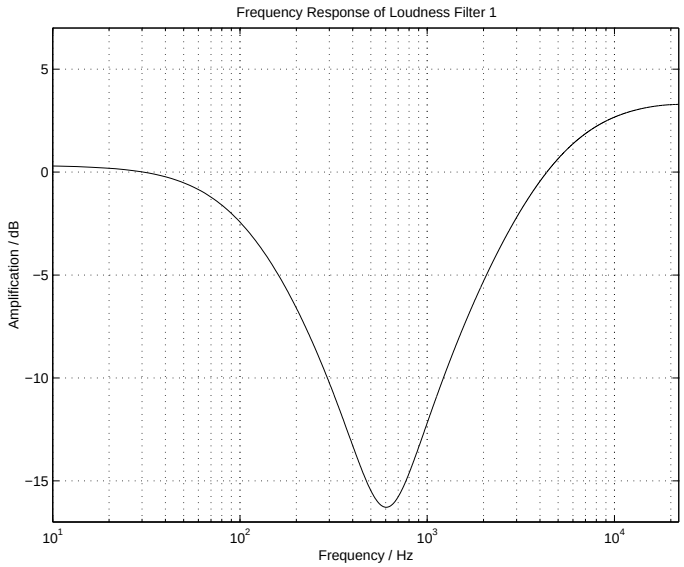

Figure 6: Filter 1 Frequency Response at 44100 Hz.

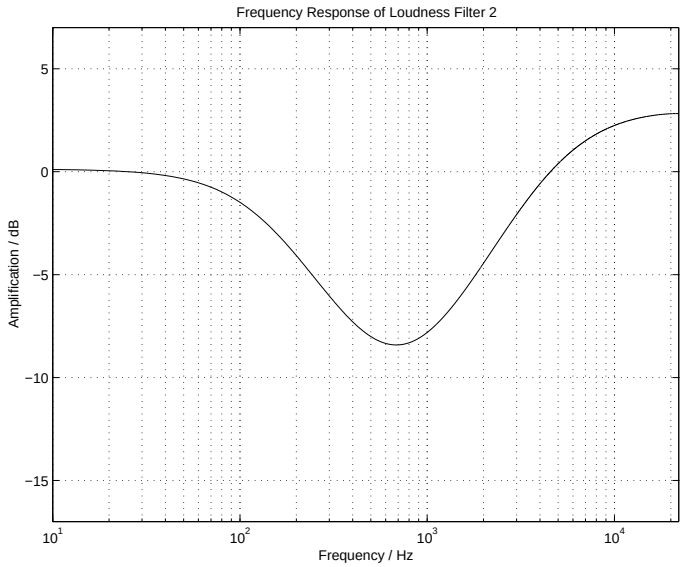

Figure 7: Filter 2 Frequency Response at 44100 Hz.

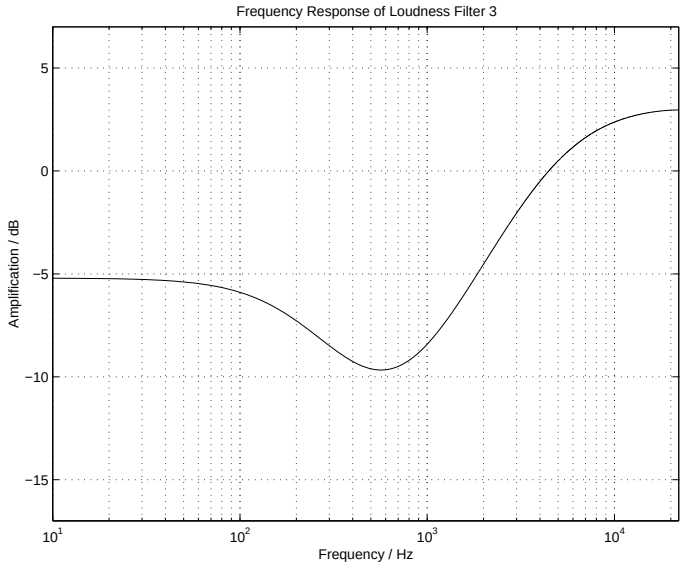

Figure 8: Filter 3 Frequency Response at 44100 Hz.

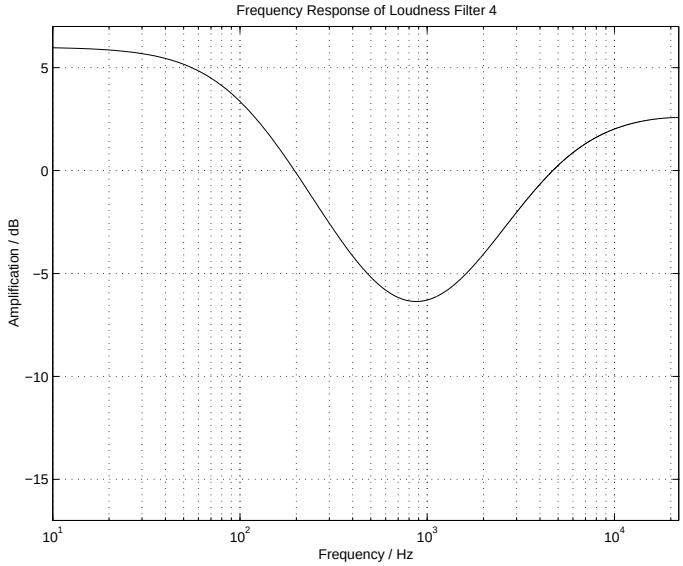

Figure 9: Filter 4 Frequency Response at 44100 Hz.

In responses for Filters 5...12, the continuous line shows the frequency response of the whole filter. The dashed line shows the FIR response, and the dashdot line the IIR response (see Chapter 4.3). Filter pairs (see Chapter 6.1) are shown on the same page.

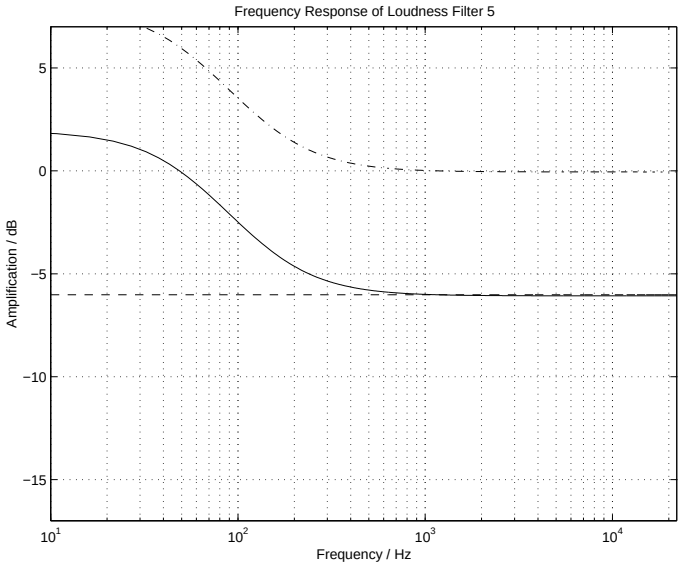

Figure 10: Filter 5 Frequency Response at 44100 Hz.

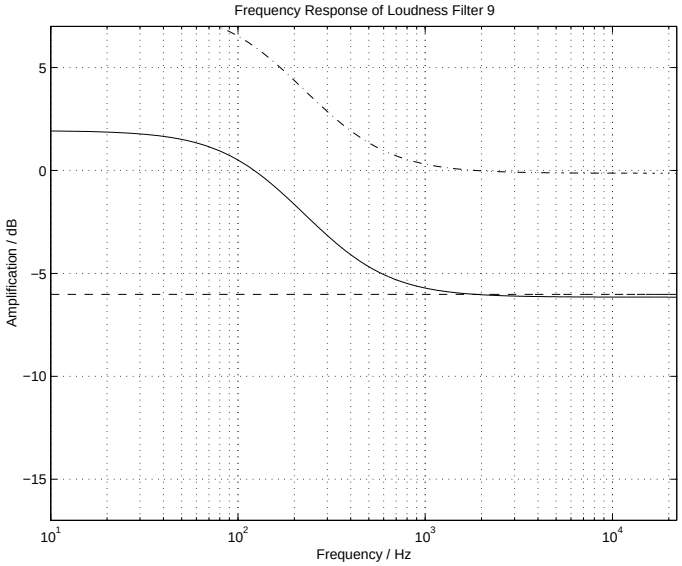

Figure 11: Filter 9 Frequency Response at 44100 Hz.

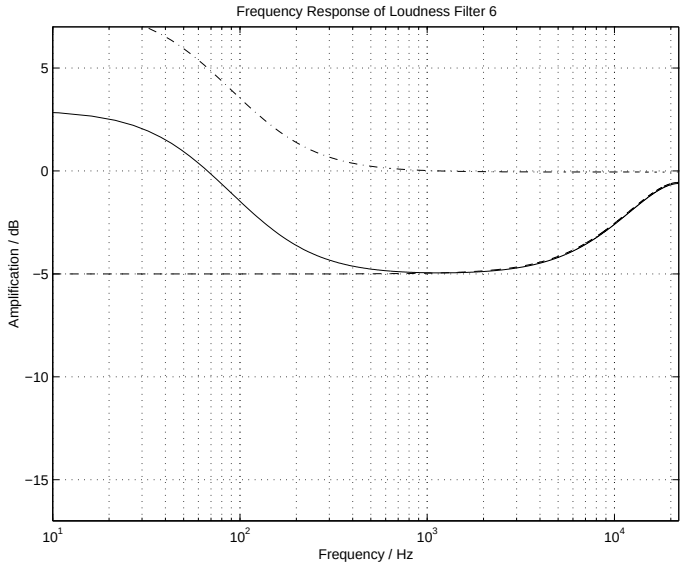

Figure 12: Filter 6 Frequency Response at 44100 Hz.

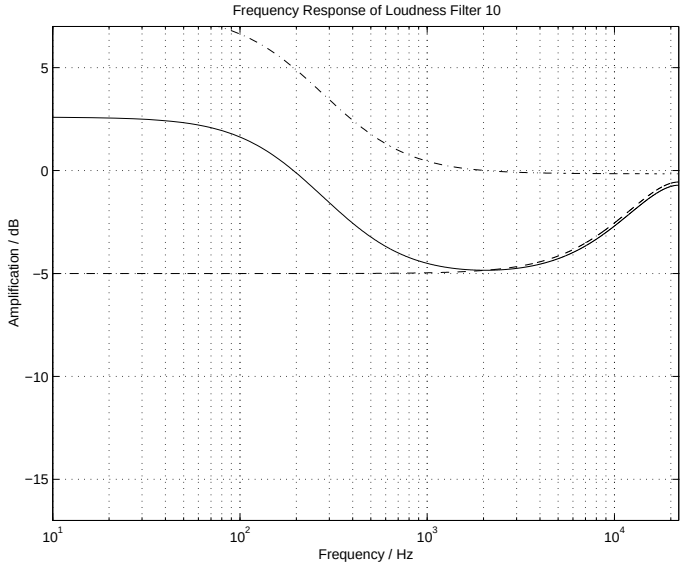

Figure 13: Filter 10 Frequency Response at 44100 Hz.

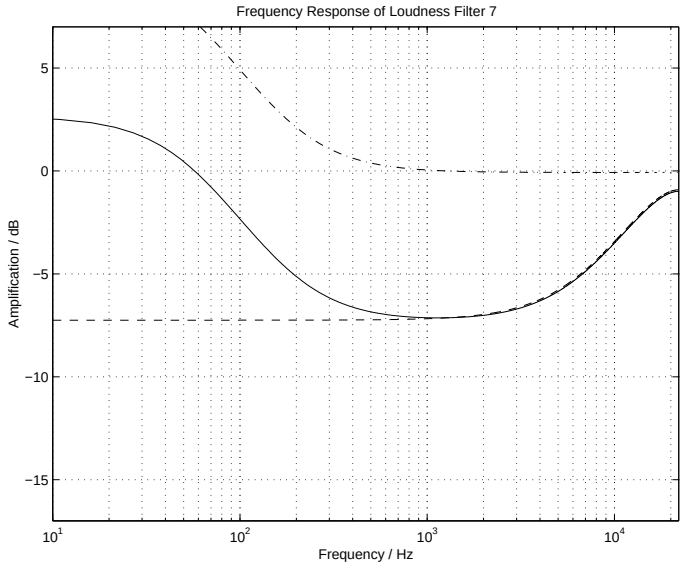

Figure 14: Filter 7 Frequency Response at 44100 Hz.

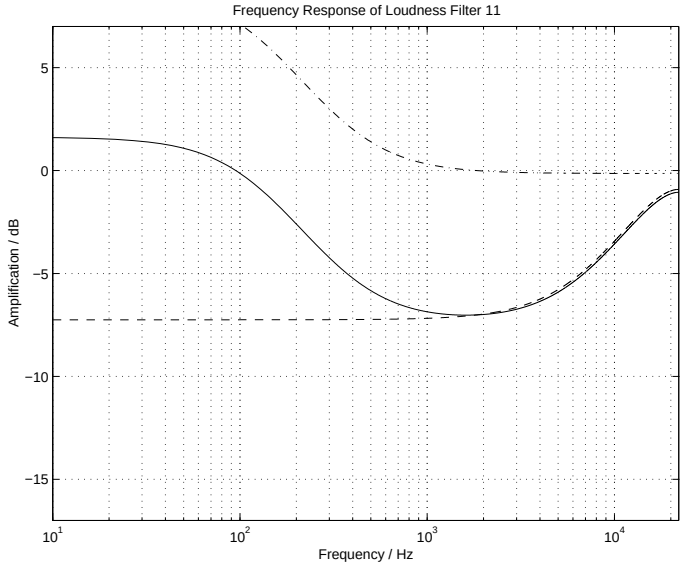

Figure 15: Filter 11 Frequency Response at 44100 Hz.

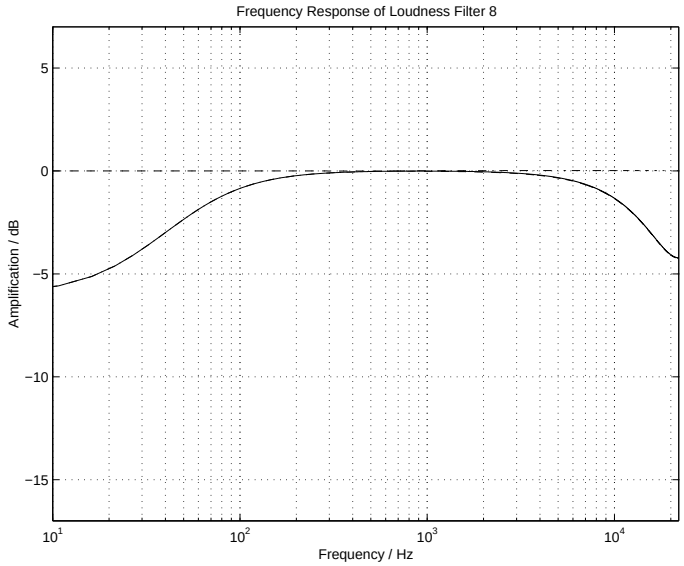

Figure 16: Filter 8 Frequency Response at 44100 Hz.

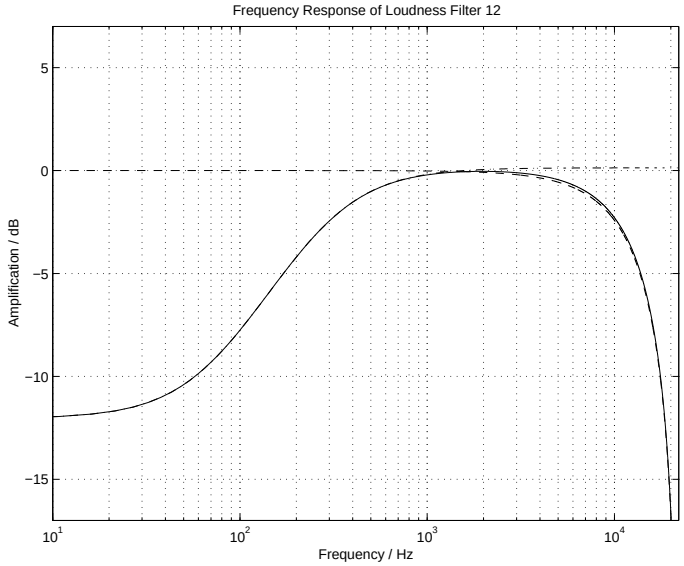

Figure 17: Filter 12 Frequency Response at 44100 Hz.

7 Usage Examples

All examples assume that VS1011/VS1002 has been reset and Loudness code has been loaded.

7.1 After Every VS1011/VS1002 Software / Hardware Reset

1. Activate Loudness code by writing 0x200 to AIADDR.
2. Reset Loudness by writing 0 to LOUD_SET.
3. Depending on the filter you want to use, initialize Loudness by writing a number between 1 and 12 to LOUD_SET. If you don't know yet which one to use, choose any of them.
4. If you don't need Loudness at the moment, write 0 to AIADDR.

7.2 To Activate Loudness

Let's assume we are playing audio at -15.0 dB of maximum volume, and we want to turn Loudness on with Filter 1.

1. Activate Loudness code by writing 0x200 to AIADDR (Initialization was already done in Chapter 7.1).
2. Because Loudness Filter 1 has an Offset of 10.0 dB, compensate for it by writing -5.0 dB to the VS1011/VS1002 volume register SPI_VOL.
3. Read Chapter 7.5 for further details on how to feed an MP3 file.

7.3 To Deactivate Loudness

Let's assume we are playing with Filter 1 as set in Chapter 7.2, and we want to turn Loudness off.

1. Reset Loudness by writing 0 to LOUD_SET.
2. Deactivate Loudness code by writing 0 to AIADDR.
3. Compensate volume offset by writing -15.0 dB to the VS1011/VS1002 volume register VOL.

7.4 To Change Loudness Filter

Let's assume we are playing with Filter 1 as set in Chapter 7.2, and we want to switch to Filter 2.

1. Initialize Loudness filter 2 by writing 2 to LOUD_SET.
2. Because original volume (before Loudness) was -15.0 dB and Loudness Filter 2 has an offset of 5.0 dB, write -10.0 dB to the VS1011/VS1002 volume register VOL.

7.5 While Feeding an MP3 File

Let's assume we are running Filter 1, which we activated in Chapter 7.2. In this case, do the following:

1. Feed the first second / 10 KB / other convenient piece of MP3 data.
2. Resend the Filter 1 activation command. This will reactivate the filter just in case VS1011/VS1002 has reset itself after some bad data. Sending this command repeatedly will *not* cause audible clicks or other distortions.
3. Repeat from step 1 until the audio file is over.

8 Loudness Distribution Package

8.1 What Do You Want to Do with Loudness?

8.1.1 Running Loudness

If you are only interested in running the Loudness program, you need only to familiarize yourself with the .cmd files (Chapters 8.2.3, 8.3.12 and 8.3.14).

8.1.2 Modifying Loudness

If you are interested in modifying and subsequently recompiling Loudness, you first have to get VSKIT from VLSI Solution. Among with VSKIT you will also get *gmake*. In addition to this, you have to get a working *perl* interpreter, version 5 or higher.

8.2 File Formats

8.2.1 .a Files

Assembly listings of compiled C files.

8.2.2 .bin Files

Compiled and linked programs on COFF format.

8.2.3 .cmd Files

The format used in .cmd files is as follows:

```
W 2 n y
```

n is the SCI register number to write to (in hexadecimal)

y is the 16-bit value to write to the SCI register (in hexadecimal).

Example:

```
W 2 7 8200
```

This command loads value 0x8200 to VS1011/VS1002's register 0x7 (WRAMADDR), and thus tells that program code is to be loaded to address 0x8200 - 0x8000 = 0x200 (see Chapter WRAMADDR (W) in the VS1011/VS1002 Datasheets).

8.2.4 .o Files

Compiled but not yet linked COFF object files.

8.2.5 .pl Files

Perl program files. You need a perl interpreter to run these files (version 5 or higher).

8.3 Individual Files

8.3.1 docs/loudness.pdf

This file, describing the Loudness software.

8.3.2 src/Makefile

A Unix Makefile, which tells the dependencies between files and instructions on how to compile the files to working programs. VSKIT includes a compilation of *gmake*, which makes it possible to compile files automatically. Run *gmake* without any arguments to compile everything.

Remember to read the Makefile, as you will have to make some changes to it depending on where you have installed VSKIT libraries.

By running *gmake testa* you may run the test version loud.bin (see Chapter 8.3.6) of Loudness.

8.3.3 src/actloud.cmd

Command file that activates the Loudness software.

8.3.4 src/c.s

Minimal startup code that sets stack pointer and jumps to main().

8.3.5 src/input.txt

Example stereo input file for loud.bin (see Chapter 8.3.6).

8.3.6 src/loud.bin

Compiled and linked version of Loudness test bench. This is a standalone program compiled from *c.s*, *loudasm.s* , *loud.c* and *loudctl.c*. It can be run in the *VSSIM* software simulator, which is provided in VSKIT.

8.3.7 src/loud.c

C source code for loudness. The main function Filter() in this C source file is not used anymore, and it's replaced with corresponding a Assembler function in *louda.s*. Thus, the C version is commented out. It is, however, still included in this C file as an example of how the Assembler function came to be.

8.3.8 src/loud.h

Definitions and prototypes for Loudness.

8.3.9 src/loud.scr

A script command file for *VSSIM* to load and simulate Loudness.

8.3.10 src/louda.s

Assembler equivalent for the Filter() function in *loud.c*. This replacement is highly optimized and better suited for the VS_DSP[4]environment than its C counterpart.

8.3.11 src/loud_sa.bin

Compiled and linked version of actual Loudness program. This is a program compiled from *loud.c* and *loudctl.c* that is meant to be run on VS1011/VS1002.

8.3.12 src/loud_sa.cmd

This command file loads Loudness code to VS1011/VS1002 RAM memory. This is the file that is needed even by people who do not wish to modify and recompile Loudness.

8.3.13 src/loud_sa.omd

VSOMD (Object Module Disassembler) dump of *loud_sa.bin*. This file is used as an intermediate file between *loud_sa.bin* and *loud_sa.cmd*.

8.3.14 src/loud_start.cmd

This is an example Loudness activation command file that can be run after loading Loudness code from *loud_sa.cmd* to VS1011/VS1002's memory.

The file reads as follows:

```
# Begin by activating our code at 0x200!
w 2 a  200
# Reset filters by writing 0 to LOUD_SET
w 2 c    0
# That should be it, then!
```

To activate a loudness filter, you would need to add the following lines (example is the same as src/setfilt02.cmd):

```
# Activate filter 2 by writing 2 to LOUD_SET
w 2 c    2
```

8.3.15 src/loudctl.c

Loudness test bench control file that reads in 16-bit stereo values, calls Loudness, and writes the results to standard output.

The input file must be in ascii format, one number on a line. Samples are interleaved, and left channel comes before right channel.

8.3.16 src/mem_desc

VSKIT memory description file that is used to link the test bench program *loud.bin* to correct addresses. It is also used to simulate *loud.bin*.

8.3.17 src/mem_desc.sa

VSKIT memory description file that is used to link the actual Loudness binary *loud_sa.bin*.

8.3.18 src/omdtocode.pl

A Perl script that is used to convert *loud_sa.omd* to *loud_sa.cmd*.

8.3.19 src/setfilt[00...12].pl

Command files that activate filter 0...12.

8.3.20 src/vsmpg.h

VS1011/VS1002 definitions.

9 Contact Information

VLSI Solution Oy
Hermiankatu 6-8 C
FIN-33720 Tampere
FINLAND

Fax: +358-3-316 5220
Phone: +358-3-316 5230
Email: sales@vlsi.fi
URL: http://www.vlsi.fi/