SinED

Jarkko Vatjus-Anttila <quaid@kempele.fi>

COLLABORATORS

	TITLE : SinED		
ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Jarkko Vatjus-Anttila <quaid@kempele.fi>	August 23, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

Chapter 1

SinED

1.1 SinED v1.15 User's guide!

SinED v1.15
Shareware version
User's guide

AHI.device is copyrighted
(c) 1994-7 by Martin Blom <lcs@lysator.liu.se>

Reqtools.library is copyrighted
(c) 1991-1994 by Nico François <nico@augfl.se>
(c) 1995-1997 by Magnus Holmgren <cmh@lls.se>

1.2 Introduction

Introduction

Well, well. Another sample editor, huh? I admit it, but this time it's something different. Or at least it's supposed to be.

For a long time I've occasianally needed a sample editor and perhaps a generator for my sample engineering. Sadly there exists not a single program that would contain all features that I want to use, and if some program contained them, then it was so badly programmed that it was completely impossible to use. A perfect example of a program like that is AudioMaster. I'm completely sick'n'tired of that kind of shit and that's why I began to write my own editor. At least this time I get what I want.

So, currently SinED is equipped with powerful arsenal of sample editting effects and of course a possibility to play the samples or even edit them by hand. Effects include several commands that have been seen in other editors, but also a bunch of new ones that should have always been there. Because of the modularity of the main program the effects can always be added, or the previous effect routines can be edited or deleted with no big hazzle. Some of the effects are even written in *very* optimized assembly language, so the speed should not be a problem. At least 060 owners laugh away with SinED.

SinED can also calculate few most common waves, like sines. No more whining about that flawless sines cannot be calculated. This time they are not just good, they are excellent. As you might have already read, all calculations are made in 16bit accuracy to make sure that the quality of the samples will never drop below good.

Learn how the effects are calculated so that there will be no more those pathetic programs like AudioMaster. Use the information resources when you can!! If you are interested to see more detailed explanation over the effects, the take a look at the Audio.FAQ section. I'm sure you get some fun out from it.

AudioLab16 is probably the only sampleprogram that I've not seen at all. However, if it's half as good as the readme files say, it beats SinED anytime. I trust that not all people can afford AudioLab16, and I'm not definitiviely thinking that the output of SinED and AudioLab16 differ that much.

I agree that this text was quite a hype, but I suggest you just to try this program. You'll see what I'm talking about.

1.3 DISCLAIMER

Disclaimer:

THERE IS NO WARRANTY FOR THE PROGRAMS, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAMS "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAMS IS WITH YOU. SHOULD THE PROGRAMS PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY REDISTRIBUTE THE PROGRAMS AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAMS (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAMS TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

1.4 Supported wave types

```
Supported Wave types
--------------------

  SinED has several wave types build-in, so that you could create your
own samples, or mix them with other audio data. No more whining about
that pure sines do not exist.

Supported wave types are:
~~~~~~~~~~~~~~~~~~~~~~~~~

Sine:   If someone doesn’t know this, then I wonder what
    you’ve been doing during your math lessons. SinED
    calculates this wave into the current buffer filling
    it completely. So, before calculating the wave, you
    should reduce or enlarge the buffer to the size you
    wish the wave to be. Purity is ensured by calculating
    the wave in 16 bit accuracy with 8 decimals.

Saw:    Saw wave is calculated with linear interpolatios and
    like the rest of the waves.

Triangle: Triangular wave is like Sine, but has sharp edges.
    This wave almost looks like a sine, and sounds like a
    sine. Even a modulation result sounds the same. Note
    that when you calculate triangular waves, the buffer
    should be rather large (more than 5kb) because linear
    interpolations are not the most accurate calculations
    around and that can make the sample to click.

Box   Does this need introduction?

White noise:  Oh, boy. White noise with 16bit accuracy. This is what
    I like. The noise is calculated with C function
    RangeRand(), which means that the numbers aren’t
    actually random. Everytime you restart the SinED, the
    generator is reset and it begins to spit same numbers
    over and over again. Hell, who needs this anyway.

Clear:    Uhh, does this need introduction either?
```

1.5 Supported effects

```
Supported effects
-----------------

  Like (almost) every other so-called soundeditor has at least few of
these. In general matter, programs around only know how to boost and
perhaps how to filter the sample. This time it will be different.
SinED offers you effects that usually are equipped in programs that
need DSP or something similar too expensive. Some of the routines may
be slow, but that’s definitiviely because they need *a lot* of
calculations. As I told you, the FPU is recommended.
```

If you are a programmer, take a look at the detailed explanations about these routines. There is info about these effects in techical way described mainly from the programmer’s point of view.

Shortly the effects are:

Boost: Increases sample peaks making the sample sound clearer. Using too powerful boost, you can easily make your samples snap, crackle and pop. Note that Boost and filter are combined in SinED so there is no separate gadget for Boost. Just click on the Filter gadget instead.

Filter: Opposite to Boost. Lowering the sample peaks you can smooth the sample and make possible cracling to fade away, but on the other had: too much filter and you won’t hear anything from the sample.

Backwards: Flips the sample backwards.

Flip: Flips the sample upside down. Does not actually affect the sound at all.

Shadow: Resets user defined values from the sample to zero. Usually, especially with speak samples this effect makes them to sound like an insect but by changing the shadow parameter you can get very bizarre results.

Blur: Increases or decreases each sample value a bit. The sample gets a foggy feeling. This routine can be used with boost. If the sample doesn’t seem to be affected by the boost operation, blur it a little and try again.

Maximize: Increase the volume to maximum without breaking the sample. When maximizing, SinED displays the progress indicator two times. The first time SinED searches for the factor that the sample has to be multiplied with to achieve the loudest volume level. The second indicator then is displaying the actual maximize process.

Minimize: Decrease the volume to minimum without losing any sample peaks. Minimize is displaying too two progress indicators. No need to panic, just check out the description of Maximize.

Modulate: Modulation means that the original sample is multiplied with the sample values from another sample waveform. Usually the modulation wave is a normal sine but SinED allows you to use other waves too or even external samples. If you ask me, the modulation results sound best when using Sine waves.

Mix: Mix two samples together. Basically by calculating the averages between the samples. Currently SinED ignores the sample frequences and mix sample valuea at a time. If range is not selected then the mixing is performed from the beginning of the original sample. The size of the result buffer is equal to the original sample. This means that if the sample used in mixing is larger than the original, it will be clipped. If the range is selected then the mixing is performed on this area no matter if the mixing sample is larger or smaller than the range area.

Echo: Make echo to the sample. Repeat tells how many times the echo has to be repeated. Delay is given in samples and basically you can say that it depends on the sample frequency. The bigger frequency, the bigger delay you should use that the echo wouldn't be too fast. Decay is given in percentages and it states how many percentages next echo is from the previous one. Note that too fast echo with too small decay value will produce only blank sample because the sound fades so fast away. SinED tries to prevent this from happening,
Note: but it's not always possible. If the buffer is too small for all of the echoes you've selected, the ones that exceed the buffer limits are just ignored. If you are using rather large repeat value, it might be best to add some extra work space after the sample. Just in case...

UpSample: Shrink the samplebuffer the amount that user defines. This makes the sample sound higher although the playing frequency remains unaltered.

DownSample: Opposite to UpSample. This routine enlarges the sample buffer to make the sample to sound lower although played with normal frequency.

Cleanup: Cleanup cleans up obsolete zero samples from the beginning and from the end of the sample buffer. This effect always affects the whole buffer which means that the possible range selection is always ignored.

Volume slide: With volume slide you can change the volume level of the sample to your own custom level, but like the name says, you can make volume sweeps too. The volume level is given in percentages. So by giving start 100 and end 100 the resulting sample is the same than the original. Be carefuf with this effect. The warning that SinED tells you about clipping is not a joke. If you volume boost your sample too much, it starts to snap, crackle and pop.

Flange: The result wave of this sample is quite hard to explain. You've probably heard flaged samples before so just try this option and you'll see what's it all about. The result is calculated by mixing the waveform into itself with slightly higher frequency. The higher the flange power is, the bigger the frequency difference is and the faster the flange effect will be. The Flange in SinED is slightly different than in other similar programs,because you can create also a closing flange in addition to the opening flange that 'almost' every

Tip: decent sample program can perform. Try to do it like this. 1) Range the first half of the buffer. 2) create a flange with parameters: start 0 end 50. 3) Range the latter half of the area. 4) And create a flange with values: start 50 end 0. Here you have a cool flange sweep.

Following 'effects' can be counted in modifiers. Sometimes you see, or at least I've seen samples that have been saved in some very, very odd way. Convert and Translate are used on those samples. If you are very certain that you've really loaded a sample, but it still looks distorted, then try these. They might convert the sample into listenable form.

Convert: Convert restores the sample from unsigned into signed form and vice versa. I don't know how to detect if the sample is unsigned or not, so you have to try this one yourself.

Translate: Translate is used when 16bit sample is saved in 'Intel' -format (LSB first). Each sample value (each word, that is), is turned around to convert the waveform into 'Motorola' -format (MSB first). Again I have no idea how to detect whether the sample is saved in 'Intel' or 'Motorola' -way. Try this one if sample looks blank or similar.

1.6 Loading and saving

Loading and Saving

Shareware
version of SinED doesn't allow to save your work.
Please encourage me to continue the development of SinED and consider registering if you've not already done it.

Currently SinED handles the following samples succesfully:

- IFF-8SVX
- IFF-16SV
- WAV 8-16bit (including 12bit sub-format)
- AIFF 8-16bit
- 8bit RAW
- 16bit RAW

Since version 1.15 SinED can now import and export stereo samples too. All of the formats described above support stereo mode, and SinED can succesfully handle them all.

Following subformats are still NOT supported:

- WAV A-Law, WAV U-Law
- AIFF 32bit.

If SinED doesn’t recognize the samplefile, then you are prompted to choose if the sample has to be loaded as RAW data. As you might imagine, the RAW data doesn’t include any headers or other information that it could be identified with.

If you like to see a support for other sound formats, then tell me about it. I think the mentioned are the the most common ones and SinED SinED will do just fine with them.

1.7 Author information

Author Information

```
  To get in touch just drop me a mail. Don’t hesitate, do it!!
                   ~~~~~~~
SMail:  Jarkko Vatjus-Anttila   EMail: quaid@kempele.fi
  Linnukkatie 2
  90450 Kempele
  Finland

WWW:  http://www.kempele.fi/~quaid
```

Note that my EMail will expire on 25th of May 1997, but before that feel free to contact me. The SMail address works just fine after that too. When contacting, get ready for a small delay. This is due to my military service.

1.8 Thanks

Thank yous:

Here is a list of people who deserve thanks. Thanks for beta testing, thanks for giving ideas, suggestions and flames. Thanks for the feedback.

```
Ville Helin   Very special thanks for testing and encouraging.
      Check out the very cool Gameboy emulator

               Wzonka-Lad
               made my him.
Samu Nuojua   Thanks for
               SNMA
              , the best macro assembler
```

around for Amiga. All assembly parts of SinED
were compiled with SNMA V2.108.
Kari Taskinen   Thanks for testing and giving the good ideas.
It may be that I now understand the whole
program better from users point of view.
Thomas Wenzel   Thanks for the help with exotic sample
formats. Thanks especially for the sources.
All beta testers You know who you are. You’ve been very helpful.

Big thanks also to the following gentlemen:

AHI.device is copyrighted
(c) 1994-7 by Martin Blom <lcs@lysator.liu.se>

Reqtools.library is copyrighted
(c) 1991-1994 by Nico François <nico@augfl.se>
(c) 1995-1997 by Magnus Holmgren <cmh@lls.se>

1.9 Shareware notice:

SinED is shareware!:

Yes, SinED indeed is shareware. I’m strongly encouraging you to register this program. By doing so you encourage me to continue the development of SinED. Support the shareware concept and register your quality software today!

If you are registering alone:

If you are registreing SinED for yourself alone you have two possibilities. Pay the registration fee in Finnish marks (FIM) which costs 80 FIM or send your own currency worth of 100 FIM. The second payment method is a little bit higher because I’ve to change the money myself and that costs 20 FIM each time. 100 FIM is approximately 20 USD (US Dollar) or 35 DEM (Deutsche Mark) or 15 GBP (Pound sterling). Fill the registration form and send it with your payment to
me
.

If you are registering in group:

If you are registering SinED in group which has two or more persons you may forget the extra 20 FIM I need in order to exchange to money. When registering in group, all of the persons have to fill the registration form and the filled forms have to be sent to me in the same envelope or package to avoid any misunderstandings. The keyfiles are then send to each of you separately, each with own disk. The prices go:

Registrations:	Cost each:	Cost total:
2	75.0 FIM	150 FIM = 30 USD = 50 DEM = 20 GBP

3	66.6 FIM	200 FIM = 40 USD = 65 DEM = 25 GBP
4	62.5 FIM	250 FIM = 50 USD = 80 DEM = 30 GBP
5 or more	50.0 FIM	>250 FIM

1 USD = 5 FIM, 1 DEM = 3 FIM, 1 GPB = 8 FIM.

When registering get ready for a small delay because of my military service. I'm able to read my mail almost every weekend. All registrations will be processed with highest priority, though.

Remember that when registering SinED, you register it to your own personal use! It may not be used by companies or other similar large groups.

Remember to fill, print and send the Registration form .

What you get when you register:

- A keyfile that enables you to save your work.
- Latest version of SinED if you don't already have it. Remember to state the version you are currently using. However, the latest released SinED can be found almost always from Aminet.
- A free disk. Registrations are made with SMail, so I'll buy the disk on which the latest SinED and the keyfile is then sent to you.
- Full right to use the results that you've created with SinED anywhere you like.

Note these facts when you register:

- My EMail account expires on 25th of May 1997. After that I'm available only via SMail.
- Due to my military service, the registration can delay. I try to read my SMail every weekend, but that's not always possible.
- And due to my military service again, I'm able to accept ONLY cash as payment.
- Place the keyfile in PROGDIR:

Send registrations to following address:

Jarkko Vatjus-Anttila
Linnukkatie 2
90450 Kempele
Finland

Thank you.

1.10 Gadget information

Gadget information

SinED has thousand gadgets that do several different things. If you

```
have difficulties in figuring out which gadget does what and so on,
just peek the list below:

Sample window:
  Cut   Cuts the marked part from waveform and copies
      it in the copybuffer. The marked part is
      removed from the waveform.
  Copy    Copy does the same than Cut but is leaves the
      original waveform unaltered.
  Paste   Paste operation pastes the contents of
      copybuffer to the original waveform. You may
      select whether you want to overwrite or insert
      the data.
  Swap    Swap swaps the contents of copybuffer and the
      mainbuffer.
  Zm in   This one zooms in to the area you have
      marked. If the marked area is smaller than 64
      bytes, SinED will use 64 byte range instead.
  Zm out    Zoom out enlarges the view a bit if you had
      previously zoomed in.
  All   All gadget displays the total waveform.
  Left/Right/both This gadget lets you to select which channel
      of the sample is affected when you edit it.
      If the sample is in mono mode, this gadget has
      no effect. Note too that Upsample, Downsample
      and Cleanup are not affected by this switch.
      If the sample is in stereo -mode, both of the
      channels are *always* altered!

Playsample window:
  Play waveform This gadget playbacks the whole waveform
      without displaying the progress meter. if the
      sample is in stereo mode, both channels are
      playbacked.
  Play display  This gadget plays the current waveform that is
      displayed. The progress meter is visible too.
  Play range  This playbacks the selected range.
  Stop playback Just what it says.
  Play L. wavefrm Plays the left channel waveform in total.
  Play L. display Plays the left channel display.
  Play L. range Plays the range you've selected for left
      channel.
  Play R. wavefrm \
  Play R. display |- These three do the same for Right channel.
  Play R. range /

Drum generator:
  Gadget matrix These gadgets let you to describe the sample
      sequence you want SinED to calculate. Each
      horizontal line presents one sample sequence
      and each vertical line one time interval.
  Delay   Lets you to define the speed factor for the
      drummachine mixing process.
  Calculate This one commands SinED to calculate the
      current sample sequence.
  Clear   Cleans up the sequences you have entered.
```

Effect Window:
See descriptions in
effects
page.

Wave Window:
See descriptions in
waves
page.

1.11 History

History list:

V1.15 14th July 1997

- SinED now uses shared IDCMP port for all of the windows. If you have experienced any unexpected delays or slow response from the windows, the troubles should be gone now. (The delay problem seemed to be a 060 specific problem.)
- STEREO samples are now supported! I dropped the plans for surround mode, because only AIFF supports surround modes. Mail me if you prefer to see support for surround samples.
- Preferences window added. SinED is again a bit more configurable.
- All mixing is now made in 32bit accuracy. This prevents all possible sample losses during mixing operations. Following effects are affected: Mix, Echo, Mix, Modulate and DrumMachine.
- Some effects are assembly optimized. I've taken special care that the effect loops fit in caches to be as fast as possible.
- Long waited ENVELOPE requester is here!! Envelopes can now be used with the most of the effects. Just try it.
- Bug fixed in Trianle routine. I hope it finally works.
- In conclusion this version is quite major release. New and fantastic bugs may occur.

V1.10 24th May 1997

- Uh, the 1.01 was quite buggy, sorry... :)
- Recompiled. It seems that V1.15 had some troubles that prevented it from running in certain systems. Hopefully the problems are gone.
- Mix didn't work with external samples. Fixed.
- Drumgenerator rewritten and several bugs fixed. DrumGenerator is now a SinED module, which means that the internal multitasking continues even the DrumGenerator window is open. Sample sequences are stored in memory and even the drumsamples are not flushed on exit. You can close the window without hesitation and reopen it anytime to continue the work with it. Much more user-friendly.
- Due to modularity, you may now use the playsample window to control the playback in the DrumMachine. Frequency and volume can be altered, and the sample playback can be stopped.
- Screen refresh rate detection could fail in some cases, especially with machines with GFX card. This should be fixed now. Statistics window now shows the detected rate with CPU and possible FPU.
- Filter effect changed from linear to logaritmic. It needs more calculations but is far more efficient.

- A lot of small modifications and bug fixes made. SinED should be more reliable now.

V1.01 15th May 1997.

- First public aminet release.
- Support for asl.library removed and reqtools.library is used instead. Version 38 or above is required.
- Playback is now handled through ahi.device. Version 4 or above is requred. Note that only include driver is for PAULA. If you own a special sound card, download the ahi.package and install the drivers there.
- Added a background picture. If you prefer not to see the picture, just delete or rename it.
- *A lot* of small modifications and bug fixes since last version.

V1.0 (Several versions from September 1996 to April 1997)

- More or less beta versions were used by certain persons. Only for internal use.

1.12 Source Details:

Fancy Details:

SinED is completely written in C. In the beginning it was compiled with GNU C 2.7.0.1 but later I upgraded my compiler to StormC V1.1. The source code consists of 5000 lines of C code and about 100 lines of assembly code excluding header and include files. The total compilation with all possible optimizations take a bit less than 10 minutes and a bit less than 10 megabytes of memory.

	Source size:	Object size:
DrumMachine.c	9395	5716
Edit.c	12562	7548
Effects.c	22479	10484
FileHandling.c	18063	8508
Graphics.c	4519	2820
Miscellaneous.c	9899	5200
PlaySample.c	7048	3292
Preferences.c	3757	1932
Requesters.c	7731	3912
SinED.c	23063	25080
WaveRoutines.c	5116	2348
WindowRoutines.c	16190	10344
Defs.h	12714	
GlobalConstants.h	1974	
Prototypes.h	4661	

1.13 Wzonka-Lad

```
  This is the .readme file of the Wzonka-Lad. Download it now!!
  -------------------------------------------------------------


Short:    Wzonka-Lad - Gameboy emulator v0.62.
Uploader: vhelin@cc.hut.fi (Ville Helin)
Author:   vhelin@cc.hut.fi (Ville Helin)
Type:     misc/emu
Requires: mc68020 or better and 3MB memory.
Version:  0.62
Replaces: misc/emu/wzonka-lad*

A new 100% asm coded Gameboy emulator. Wzonka-Lad is still in its early
stages, but many games are playable.

                            v0.62 (05-Apr-97)
```

1.14 Samu Nuojua's Macro assembler

```
  This is the .readme file of the SNMA. Download it now!
  ------------------------------------------------------


Short: 680x0/6888x amiga macro assembler
Type:  dev/asm
Author: snuojua@cc.helsinki.fi (Samu Nuojua)
Uploader: snuojua@cc.helsinki.fi (Samu Nuojua)
Replaces: snma-2.02.*

 SNMA is freeware conditional 680x0/688x macro assembler for  the  Amiga.  It
 supports  the  most  common  directives, generates Amiga object-files and/or
 executables. It is used mainly from the  shell.  Arexx  port.   AmigaOS  2.0
 and greater only. AmigaGuide docs only. This is version 2.03.

 New features in 2.x:
                 - Options setting program, SnmaOpts
                 - some minor changes and additions + bugs fixed

 Changes since 2.02:

    SNMA
        o One bug fixed from the option file loading, which caused random
          crash.
        o Bitfields can be defined like "<ea>,{o:w}".
        o Some movec problems disappeared (this one is little mysterious,
          since I didn't touch the whole movec thing).
        o In ARexx mode, snma could flag some addressing modes as invalid,
          because certain data structures had spurious values left from
          the previous assembly.
        o ARexx SET command works again.
        o In (BD,An,Xn) register equates, which started with z, were not
          recognized.
        o QUIET switch disables now all output to Output().
        o If expression ended to operator, snma read one long from the address
```

```
          which was fetched from the string (one indirection too much). This
          could cause buserr exception in 68000, depending on value it read.
        o @ character is now normal symbol character, by default.
          @ character in the beginning of symbol meant octal number specifier.
          Since SAS/C uses it in the beginning of __regargs function names,
          its behaviour has been changed. Use OCT commandline toggle to
          allow octal numbers. There was also a bug, which caused no error
          telling in @symbol case, when it was actually coded as octal number
          - unsuccessfully.

 2.02 fixes the following bugs since 2.00:

    SNMA
        o BSS sections didn't honor mem type bits.
        o Sometimes pc-relative addressing modes allowed small data references.
          (if small data references were allowed).
        o RTM was coded badly.
        o Forcing Base Displacement to word or long didn't work always.
        o  " macro <macroname>" finally works.

    SnmaOpts
        o Some checkbox gadgets were mixed with each other
        o If started from project icon, doesn't try to read project file
          as option file if it doesn't exist.
```

1.15 Requirements

```
  Hardware and software requirements:
  -----------------------------------

  SinED actually requires very little. Needed software are:

  - Kickstart v2.04+
  - Reqtools.library V38+ (included, copy to Libs:)
  - Iffparse.library V37+ (included, copy to libs:)
  - AHI.device V4+ (included, copy to Devs:)

  - Gadtools.library V37+ (not included)
  - Mathieeedoubbas.library V37+ (not included)
  - Diskfont.library any version (not included)
  - Utility.library (not included)
  - SetPatch

  I've included the latest AHI at this time in this package. Only
  included driver is for Paula. If you own a special sound card, just
  download the AHI package and install the drivers from it. Use the
  prefs program 'AHI' to configure the device. SinED uses the
  AHI_DEFAULT_UNIT, which is unit 0, and the default settings you've
  defined with the AHI prefs program. The latest AHI can be always found
  from:

  http://www.lysator.liu.se/~lcs/ahi.html

  Required hardware are:
```

- Amiga.. :D (I hope not for long, though. I’ve planned to recompile SinED under Linux.)
- About 300kb of any memory for the program itself. Samples can then take awfully lot of memory but that’s another story.
- SinED881 requires 020+ CPU and maths co-processor. 040 and 060 will also do just fine.
- SinEDIEEE is compiled to run on mc68000 without maths co-processor so no special hardware is needed.

SinED was tested with following configurations:

```
A1200/mc68020,       2MB Chip,            KS v39.43, WB v39.29
A1200/mc68030/68882, 2MB Chip, 16MB Fast, KS v40.68, WB v39.29
```

I don’t know my betatesters’ exact configurations, but SinED is known to run with mc68040 and mc68060 without problems.

1.16 Distribution

Distribution conditions:

Shareware
version of SinED is freely distributable if:

- All files are included in the package unaltered.
- Only minor payment to cover copying costs is allowed. Absolutely no more than $5 USD

SinED can be included in coverdisks and CD-ROMs but if that’s done, I’d be grateful to have a free copy of that magazine/coverdisk/CD-ROM. Check out my
address
.

SinED V1.15 is copyrighted (c) 1996-7 By
Jarkko Vatjus-Anttila
AHI.device is copyrighted
(c) 1994-7 by Martin Blom <lcs@lysator.liu.se>

Reqtools.library is copyrighted
(c) 1991-1994 by Nico François <nico@augfl.se>
(c) 1995-1997 by Magnus Holmgren <cmh@lls.se>

1.17 Drummachine instructions:

Drummachine instructions:

This is heavy. I wonder what you can do with this beast. Well, you said it: drumfills. Yes, this is not about creating drumsamples from scratch, as some of you might already think there. What you need in order to use Drumfill generator is decent drumsamples and a good sence of rhythm. With your instructions SinED calclates a drumfill for you, which can then be used in music modules or similar.

Sample sequences:

SinED allows you to mix 8 drumsamples at a time. Sample sequences tell SinED when each of the samples has to be played. You don’t need to worry how rapidly you use the sequences. SinED calculates the final drumfill in a way that no sample fades away because of another. All drum samples are audible from the beginning to the end.

So, the delay factor in the upper right corner of the window tells you how many bytes is the delay between one sample sequences. This depends on the length of your drum samples. Let’s say that the length of your bassdrum is about 4500 bytes. Then you can imagine that if you leave each time three empty sequence slots between the bassdrums, the delay factor should be around 4500/4 = 1200 to make sure that the bassdrum is audible in total.

If you can’t imagine that story above, just try selecting the delay factor to 1500 and then test the sample to find out the correct delay factor.

SinED allows you to use 32 sequence slots, but if you are about to create a very fast drumfill, 16 might suit you better. In this case just fill the first 16 as you would do normally and calculate the sample. Then just halve the buffer size using the length gadget in sample window and the result is exactly what you want it to be.

Note: After creating a drumfill, you may want to maximize or volume slide it to raise the volume level to your cutsom value. The drum machine does not give maximized waveform as output.

Note2: Drumgenerator is basically just a mixing machine. If you like, you can use it to mix 8 samples at a time together.

Load and test gadgets:

With load gadget you can load a single sample for that row. Each row has its own control gadgets. If there is a star (*) next to the load gadget, it indicates that you’ve already loaded a sample for that row. If you have forgotten what the sample was, just press the Test button. Playback can be controlled with PlaySample window. The frequency and volume can be altered, and the playback can be stopped.

DrumGenerator works now in parallel with other windows. This means that if you are generating a drumfill with the generator, you don’t need to close and cancel it to do something else. Just push the window background or close it, the changes are saved in memory.

If you wish to unload a sample from a row, just press the load gadget and the cancel the requester.

1.18 General notes:

General Notes:

Here is a list of general notes on the usage of SinED. Sometimes you may find a frustrating feature and want to kill me, but before that read the list here, just in case that you might have forgotten someting.

You can consider this list also as "known bugs".

- Still SinED cannot actually be launched from WorkBench. I've included project icons and scripts and I use IconX to launch it from an icon, but that's not actually a WorkBench launch. Do NOT try to put SinED in toolmanager dock or similar and define it as WBench program. It will NOT work.

- Always when you change the version of SinED, delete the old preference files you might have. ENV:SinED.prefs and ENVARC:SinED.prefs. SinED notices if the format of prefs file is changed and ignores the incompatible version, but still by deleting the old prefs file you make sure that no troubles encounter.

- The menus probably look old fashioned with your machine, but that's excatly I've meant them to be. Somewhy newlook menus are not working with me. If I use them, I just get a result that no menus are visible, only the menuitem texts can be seen. I've no idea is the bug in my code or is there something conflicting with SinED like MultiCX, but until I resolve that, the menus stay old fashioned.

- Currently only your amount of memory is the limit for the actions. However, there is a slight possibility that you are about to edit a track that you've just sampled from your favourite CD. The sample is 16bit RAW and takes approximately 80 Megabytes of memory. Nobody has a memory of that kind. That's why I've been planning to add a feature to SinED which allows you to calculate the effects directly on the sample that lies on your harddisk without loading it into memory at all. This would make the editting of HUGE samples possible but the results would be permanent as you can imagine.

1.19 Preferences

Preference settings:

This page shows the setting possibilities that you are allowed to choose in this version of SinED. Default for all of the settings is OFF.

- DirectEdit -mode

  DirectEdit -mode is meant to give a opportunity to edit samples that are larger than your total amount of memory. When this flag is activated, none of the samples are actually loaded into memory at all. They are edited on the disk directly. As you can imagine, if the sample file is altered, the change is very permanent and cannot be cancelled. Therefore you should have a backup of the samples that you are about to edit with DirectEdit -mode.

  DirectEdit -mode is not yet fully functional and is disabled in this version of SinED.

- Display warnings

  This option tells SinED to shut up, even if it noticed something that could cause trouble. This flag should be activated, since SinED doesn't display warnings if not absolutely neccessary.

- Confirm quit

  This tells whether SinED should display "Are you sure to quit?" message or not.

- Maximize

  This flag tells SinED always to maximize the sample before saving. Especially with 8bit samples, they should be maximized before saving to achieve the best dynamics. This flag uses the maximize routine to maximize the sample, so the result is exactly the same if you used the maximize effect before saving.

- Analyze sample

  Analyzing sample means that when SinED loads a sample file, it automaticly calculates some of needed variables. Currently this doesn't do anything else than seek the highest absolute value from the sample, but it may change in the future. Highest absolute value can be used for example with maximize and minimize routines, to halve the normal time needed to calculate the effects.

  This option is not implemented yet either!

- Mono -> Stereo

  This flag tells SinED to convert mono samples always to stereo when imported to SinED. The conversion can be done later manually too.

- No Gfx

  This flag tells SinED whether the background picture is needed or not. If you don't like it or you have troubles with it. Just disable it.

-