

ahiusr

COLLABORATORS

	<i>TITLE :</i> ahiusr		
<i>ACTION</i>	<i>NAME</i>	<i>DATE</i>	<i>SIGNATURE</i>
WRITTEN BY		August 27, 2022	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1	ahiusr	1
1.1	ahiusr.guide	1
1.2	ahiusr.guide/Overview	2
1.3	ahiusr.guide/Distribution	3
1.4	ahiusr.guide/Donations	3
1.5	ahiusr.guide/System description	4
1.6	ahiusr.guide/AddAudioModes	4
1.7	ahiusr.guide/AHI	5
1.8	ahiusr.guide/Menus	6
1.9	ahiusr.guide/Project Menu	6
1.10	ahiusr.guide/Edit Menu	7
1.11	ahiusr.guide/Settings Menu	7
1.12	ahiusr.guide/Help Menu	8
1.13	ahiusr.guide/Pages	8
1.14	ahiusr.guide/Mode settings	8
1.15	ahiusr.guide/Advanced settings	9
1.16	ahiusr.guide/AHI-Handler	10
1.17	ahiusr.guide/System Files	11
1.18	ahiusr.guide/The Drivers	12
1.19	ahiusr.guide/The Mode Descriptors	15
1.20	ahiusr.guide/The Author	16
1.21	ahiusr.guide/The Man	16
1.22	ahiusr.guide/The Myth	17
1.23	ahiusr.guide/The Concept	18
1.24	ahiusr.guide/Acknowledgments	18
1.25	ahiusr.guide/Contributors	19
1.26	ahiusr.guide/Concept Index	20

Chapter 1

ahiusr

1.1 ahiusr.guide

AHI User's Guide

For AHI version 4.20. Document version 4.21 (1999-03-28).

Copyright (C) 1994-1999 Martin Blom

The latest release of AHI can always be found at
<http://www.lysator.liu.se/~lcs/ahi.html>.

Overview

Brief introduction

Distribution

What you are allowed to do and not

Donations

How to show your gratitude

System description

The components of AHI

The Author

Who designed it?

Acknowledgments

Thanks, guys!

Concept Index

Concept Index

1.2 ahiusr.guide/Overview

Overview

The Amiga has always had excellent sound capabilities. In 1986, they were awesome. Today, well.... Perhaps not awesome, but still very good. The OS interface, audio.device has however never been as good as it could have been. It is tied hard to the underlying hardware, and doesn't work very well for music. This has led to a situation where most audio programs only use audio.device to allocate the audio resource, and then poke around in the hardware registers--making it next to impossible to replace the Paula chip (1).

There have been attempts to write an audio.device clone that uses a sound card instead of Paula, but so far nobody has succeeded. It is definitely possible, but the question is if it is worth the trouble--too many of the programs bang the hardware.

Entering AHI (2). AHI is a new audio subsystem, designed to be flexible, hardware independent, expandable and future safe. It is designed with real-time applications in mind. It is designed to play modules (3) and sound effects as efficient as possible, taking advantage of modern DSP-based sound cards.

Yet AHI allows applications that don't need full control over the audio hardware to share the resource, so that many different programs can play and record sound at the same time, without conflicts.

As a user you will hopefully not see much of AHI, other than the audio mode requesters. They works almost exactly like screen mode requesters.

AHI was never supposed to be the standard for hardware independent audio. It was meant as a temporary solution until Amiga Technologies delivered an official standard. However, the situation looks worse and worse for every day that passes by, and this may be all you will ever get.

----- Footnotes -----

(1) Paula is one of the custom chips, and she is responsible for the sound (and more). Unfortunately, this chip has not been updated since the very first Amiga was released.

(2) The name AHI was chosen because the functions in the system had to have a prefix, and the author couldn't come up with anything better than Audio Hardware Interface, something that he has regretted ever since. The suggested pronunciation is "atchii", as in "God bless!".

(3) Originally designed in 1986 by Karsten Obarski, modules have become a de facto standard for game and demo music. The original format has been improved many times, and many new music formats have--more or less--been derived from it, including the popular S3M and XM formats.

1.3 ahiusr.guide/Distribution

Distribution

Copyright (C) 1994-1999 Martin Blom

AHI is available as freeware. That is, it may be freely distributed in unmodified form with no changes what so ever, but you may not charge more than a nominal fee covering distribution costs. However, donations are welcome (see
Donations
).

If you use this software in a commercial or shareware software product, please consider giving the author (see
The Author
)--and
preferably each one of the contributors (see
Contributors
) too--an
original or registered version of your work. Should you want to distribute the AHI software with your own product, there is really nothing to consider, is it?

If you wish to distribute this software with a hardware product, contact the author (see
The Author
). Distribution of AHI with hardware products is not free.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

1.4 ahiusr.guide/Donations

Donations

-AHI is available for personal use without any charges. I strongly believe in free software, and I use free high-quality software tools daily. But I live, like all of us, in a real world. As a student, my income doesn't even suffice for the rent, and much less food, course

literature and entertainments. Therefore I ask, if you like this software, please consider showing your gratitude by making a small donation (see

The Author
for my address).

Martin

1.5 ahiusr.guide/System description

System description

AddAudioModes
Database maintenance

AHI
Preferences program

AHI-Handler
DOS-device

System Files
The files in DEVS:

1.6 ahiusr.guide/AddAudioModes

AddAudioModes

=====

Format

[FILES <file|pattern>] [QUIET] [REFRESH] [REMOVE] [DBLSCAN]

Template

FILES/M, QUIET/S, REFRESH/S, REMOVE/S, DBLSCAN/S

Purpose

To build and remove a list of audio modes that AHI can understand.

Path

C:ADDAUDIOMODES

Specification

AddAudioModes is used to build and remove a list of audio modes that AHI can understand. The definitions of the audio modes are

stored in DEVS:Audiomodes (see The Mode Descriptors). Normally you don't have to run this program, since ahi.device automatically reads all mode descriptors when it is used for the first time. It can, however, be useful in installation scripts.

The FILES option specifies with descriptor(s) to be added to the current mode list.

The QUIET option, if specified, will suppress error and output messages.

The REFRESH option, if specified, will scan DEVS:Audiomodes and add all descriptors found there to the current mode list.

The REMOVE option, if specified, will flush the current audio mode list from memory.

The DBLSCAN option does not have anything to do with the audio mode list. If specified, it will open and then immediately close a native, double-scan screen. On some systems using a graphic card, this will enable >28 kHz sample frequencies with the native audio. You need an appropriate monitor driver in DEVS:Monitors to make it work.

1.7 ahiusr.guide/AHI

AHI

===

Format

[FROM <filename>] [EDIT] [USE] [SAVE] [PUBSCREEN <public screen name>]

Template

FROM,EDIT/S,USE/S,SAVE/S,PUBSCREEN/K

Purpose

To specify default audio options

Path

SYS:Prefs/AHI

Specification

AHI without any arguments or with the EDIT argument opens the AHI preferences editor. The FROM argument lets you specify a file to open. This must be a file that was previously saved with the Save As... menu item of the AHI preferences editor. For example, if you have saved a special configuration of the AHI preferences editor to a file in the Presets drawer, you can use the FROM argument to open that file. If the USE switch is also given, the editor will not be opened, but the settings in the FROM file will

be used. If the SAVE switch is given, the editor will not open, but the settings in the FROM file will be saved. The PUBSCREEN option allows you to specify a public screen on which the program will open its window.

Example:

```
AHI Prefs/Presets/AHI.Delfina USE
```

loads and uses the specifications saved in the AHI.Delfina file. If the system is rebooted, the last saved specifications will be loaded.

Note that the preferences program requires either bgui.library version 41 (1) or MUI version 3.8 (2)

Menus

Pages

----- Footnotes -----

(1) BGUI is Copyright © 1996-1997 Ian J. Einman

(2) MUI is Copyright © 1992-1997 Stefan Stuntz

1.8 ahiusr.guide/Menu

Menus

Project Menu

Edit Menu

Settings Menu

Help Menu

1.9 ahiusr.guide/Project Menu

Project Menu

.....

The Project menu options let you save the editor settings to a specific file and open previously saved files.

Open...

Loads the information from a specified preset file.

Save As...

Specify the preset file in which to save the currently displayed settings. The requester provides a default file name in the Presets drawer. If you want to change it, type in the full path to a different file and select OK.

About...

Shows the credits.

Quit

Exits the editor without performing any changes (the same as the Cancel gadget).

1.10 ahiusr.guide/Edit Menu

Edit Menu

.....

The Edit menu options allow you to restore previously used settings or the default settings. The options are:

Reset to Default

Returns the editor settings to the default settings.

Last Saved

Returns the editor settings to the last settings you saved.

Restore

Returns the editor to the settings displayed when the editor first opened.

1.11 ahiusr.guide/Settings Menu

Settings Menu

.....

The Settings menu contains the Create Icons? item that allows you to save project icons representing your editor settings in the same drawer as your files. For example, if you save the specifications to the SYS:Prefs/Presets/AHI.pre file, the icon for the file appears in the Presets window. Double-click on the icon to activate the file's settings.

1.12 ahiusr.guide/Help Menu

Help Menu

.....

The Help menu's items let you view the on-line "AHI User's Guide" using AmigaGuide.

Help...

Brings up the chapter about the preferences program (this chapter).

AHI User's Guide...

Brings up the first page of "AHI User's Guide".

Concept Index...

Brings up the index of "AHI User's Guide".

1.13 ahiusr.guide/Pages

Pages

The preferences program's GUI is divided in two pages:

Mode settings

Advanced settings

1.14 ahiusr.guide/Mode settings

Mode settings Page

.....

On this page you select which audio mode to use. You can select audio mode for both low-level programs (Music unit) and other programs (Unit n) that don't require low-level audio access such as the AUDIO: device (see

AHI-Handler

), sample players etc. You can also select the sample mixing (and recording) frequency to use and how many channels you wish use (1). Furthermore, you can set three hardware properties of your sound hardware, namely the output volume, monitor volume and input gain. Finally, you can select which input and output connectors you wish to use.

The Music unit is the defaults for low-level programs. Such programs often have an audio mode requester that lets you chose an audio mode. If you chose Default audio mode from this requester, these

settings will be used. Note that the number of channels is not selectable here, it's up to the application program to decide how many channels to use.

----- Footnotes -----

(1) The more channels you select, the more sounds can you play at the same time. However, due to the nature of sound mixing, the volume will decrease as well. If you try to play more sounds at the same time than there are channels, the least important sounds will be muted until the other sounds have finished playing.

1.15 ahiusr.guide/Advanced settings

Advanced settings Page

.....

This page contains some options that should not be used if you don't understand them.

Debug level

Sets the debug level for AHI. If not None, AHI will print debug information to the serial port.

Echo

On slow processors (like anything below a MC68040) echo can take so much CPU power that it becomes unusable, and can therefore be disabled. As an alternative, echo can be done Fast, which means that the parameters will be twisted in order to gain speed. The result may not be what the composer or programmer wanted, but at least it's echo.

Surround in "Fast" modes

In modes that use multiplication tables (the so-called "fast" modes) all surround sounds have to be mixing without using tables. In order to make mixing as fast as possible, surround sounds can be forced to ordinary ones by disabling them.

Master volume

Normally, the "Master volume" feature of AHI can make the output heavily distorted if set too high. Turning "clipping" on can reduce the distortion, but will--in the current implementation--use 128 kB of extra memory. Note that the so-called "HiFi" modes are not affected by this switch (see

The Mode Descriptors
).

CPU usage limit

Some hardware drivers (but not all) can be told not to use more than a given percentage of the available CPU time in order to prevent lockups. If your mouse pointer freezes when playing music, reduce the limit slightly. If on the other hand the sound

becomes cut and distorted, you can try to increase the limit--but remember that you risk locking up the computer!

1.16 ahiusr.guide/AHI-Handler

AHI-Handler
=====

The AHI-Handler is an I/O mechanism that is used to play and record sounds. The AHI-Handler is normally mounted as AUDIO: at startup time, or later by double-clicking on its icon or by giving the following command in a Shell window: mount AUDIO: <RET>.

The DOSDriver entry is:

```
Handler          = L:AHI-Handler
Stacksize       = 4096
Priority         = 5
GlobVec         = -1
```

When the device is mounted, you can read from the device to record and write to it to play. Options can be given like this:

```
"AUDIO:PRIORITY=1 VOLUME=50"
```

All slashes (/) in the name will be translated to spaces. Thus, if you use slashes instead of spaces, you don't have to use quotes around the name:

```
AUDIO:PRIORITY/1/VOLUME/50
```

The full template for reading is:

```
B=BITS/K/N,C=CHANNELS/K/N,F=FREQUENCY/K/N,T=TYPE/K,L=LENGTH/K/N,
S=SECONDS/K/N,BUF=BUFFER/K/N,UNIT/K/N
```

The full template for writing is:

```
B=BITS/K/N,C=CHANNELS/K/N,F=FREQUENCY/K/N,T=TYPE/K,V=VOLUME/K/N,
P=POSITION/K/N,PRI=PRIORITY/K/N,L=LENGTH/K/N,S=SECONDS/K/N,
BUF=BUFFER/K/N,UNIT/K/N
```

BITS can be one of 8, 16 or 32. CHANNELS can be either 1 for mono or 2 for stereo. The FREQUENCY is in Hertz, TYPE is one of SIGNED, AIFF or AIFC. VOLUME ranges from 0 (silence) to 100 (full volume), and POSITION ranges from -100 (far left) via 0 (center) to 100 (far right). The PRIORITY can be from -128 to 127 (unstoppable). LENGTH is how many bytes you wish to read or write, and SECONDS is the same, but in seconds instead of bytes. The BUFFER size is specified in bytes. Note that two buffers are always used, which means that the memory usage will be two times BUFFER. UNIT selects which ahi.device unit to use.

The default options for reading are BITS=8 CHANNELS=1 FREQUENCY=8000

TYPE=SIGNED LENGTH=very-very-much BUFFER=32768 UNIT=0.

The default options for reading are BITS=8 CHANNELS=1 FREQUENCY=8000
TYPE=<none> VOLUME=100 POSITION=0 PRIORITY=0 LENGTH=very-very-much
BUFFER=32768 UNIT=0.

If TYPE is not specified, the default behaviour is to identify the data stream as IFF-AIFF or IFF-AIFC. If so, the default values of BITS, CHANNELS, FREQUENCY and LENGTH will be taken from the file. You can still override them if you wish. If the stream could not be identified, the data format is assumed to be SIGNED.

Both when reading and writing the sample rate will be converted on the fly to what the underlying hardware is configured to. Normally this is not a big problem when writing, but the quality when reading leaves quite a lot to wish for, since no low-pass filters are used.

Example 1:

```
copy Louise.AIFF AUDIO:
```

plays the file Louise.AIFF.

Example 2:

```
copy AUDIO:SECONDS/10/TYPE/AIFC/B/16/F/44100/C/2 sample.AIFC
```

records 10 seconds of audio and stores it in the file sample.AIFC as uncompressed IFF-AIFC, 16 bit stereo at 44.1 kHz.

1.17 ahiusr.guide/System Files

System Files

=====

AHI uses a set of hardware drivers for each sound card. This means that it's easy to add support for new sound cards as they appear. At the time of writing, the following sound cards are supported:

- * Aura (sampling only)
 - * Clarity (sampling only)
 - * Concierto
 - * Delfina DSP and Delfina Lite
 - * DraCo Motion
 - * Maestro Pro
 - * Melody MPEG
 - * Paula (the built-in audio)
-

- * Prelude
- * Toccata
- * Wavetools

The Drivers

The Mode Descriptors

1.18 ahiusr.guide/The Drivers

The Drivers

The hardware drivers themselves are located in the DEVS:AH1 drawer, and are named as <name>.audio. They are actually libraries, in spite of being located under the DEVS: assign, and will be flushed out from memory when not in use and the system needs more RAM. Many of the drivers require additional files; see below. These extra files are not delivered with AH1.

concierto.audio

Requires concierto.library.

delfina.audio

Requires delfina.library version 4 or greater (1).

maestropro.audio

Requires maestix.library version 40.10 or greater (2). For more information about this driver as well as the most recent version of maestix.library, please visit the author's WWW page (3).

melody.audio

Requires melodympeg.device version 1.40 or greater (4).

paula.audio

On startup, the file ENV:CyberSound/SoundDrivers/14Bit_Calibration is read and used for the 14 bit DAC calibration.

The 14 bit modes cannot be used for recording, but the 8 bit modes supports both generic parallel port samplers as well as both the Aura and Clarity samplers.

The environment variable AH1paulaFilterFreq is checked every time playback starts, and should be set to a frequency in Hertz. If the mixing frequency is higher than this value, the internal low-pass filter will be turned off. If it is lower, the filter will be activated. The default is 0 Hz, which means that the filter will always be turned off. Example:

```
SetEnv AHIpaulaFilterFreq 16000
Copy ENV:AHIpaulaFilterFreq ENVARC:
```

The variable `AHIpaulaSampleLimit` is also checked. This variable controls how the driver should handle mixing frequencies greater than 28 kHz, which is the limit of the hardware when using 15 kHz screen modes (PAL, NTSC, Euro36). If the current screen mode is a VGA (31 kHz) mode, the driver allows frequencies up to 48 kHz. Normally, the driver checks the current screen mode, and decides if the higher mixing frequencies should be available or not. By setting this variable, you can control that decision. If set to 0, the frequency will always be limited to 28 kHz and if set to 1, there will never be any limit. Example:

```
SetEnv AHIpaulaSampleLimit 1
Copy ENV:AHIpaulaSampleLimit ENVARC:
```

This will disable any screen mode checking, and will always allow up to 48 kHz in the mode requesters.

```
Delete ENV:AHIpaulaSampleLimit
Delete ENVARC:AHIpaulaSampleLimit
```

This will turn on the screen mode checking again.

Please note that this 31 kHz screen mode is not necessary the screen mode you're seeing on your monitor. If you're using a graphic card, you must force the Amiga video signal to 31 kHz. CyberGraphX users might want to try this command (see

```
AddAudioModes
for more information):
```

```
AddAudioModes DBLSCAN
```

Picasso 96 users just need to set the `Picasso96/AmigaVideo` variable to 31kHz:

```
SetEnv Picasso96/AmigaVideo 31kHz
```

Because of incorrect hardware documentation, there is great confusion about which hardware channels are sent to the left speaker, and which are sent to the right. `paula.audio` uses the correct order (right, left, left, right) but many other programs don't. The `AHIpaulaSwapChannels` variable was added to let the user decide if the correct or incorrect behaviour should be used. In not present or set to 0, the correct behaviour is used. If set to 1, the left and right channels will be swapped.

By setting the `AHIpaulaFakeMixFreq` variable to 1, you can make `paula.audio` not report the actual mixing frequency used, but rather exactly the frequency that the program asked for. The default, 0, will report the nearest possible mixing frequency that the Paula sound chip can use.

Why would anyone want this, you may ask. Well, by setting the variable to 1, you will make `paula.audio` behave exactly like

filesave.audio, which can be important if you are making music that you will later render and put on a CD, for example. Be warned, however, that setting this variable to 1 can make the sound produced sound a little false (but not when rendered, of course)!

Finally, the variable AHIpaulaBufferLength controls the minimum playback buffer size to use. Because of the limited Chip RAM bandwidth, a MC68060 CPU might run into trouble when using the default minimum buffer size (0). By setting this variable to 1024, for example, you will reduce the number of interrupts caused and increase the number of samples transferred each time to at least 1024 samples. But take care! Setting this variable too high will cause long periods with multitasking disabled.

toccata.audio

Requires toccata.library version 12 or greater (5). This driver also reads the environment variables AHItoccataNoTask and AHItoccataIrqSize. If AHItoccataNoTask is set to 1, all mixing will be done in a Software Interrupt which means the sound output will not suffer when multitasking is turned off. The back side is that it requires a faster CPU. Much faster. Only use this option as a last resort. Example:

```
SetEnv AHItoccataNoTask 1
Copy ENV:AHItoccataNoTask ENVARC:
```

AHItoccataIrqSize specifies the number of bytes transferred to the card each interrupt and defaults to 512. It must be one of 32, 64, 128, 256 or 512. If you encounter problems with serial port hardware, you might want to set this variable to a lower value than the default.

Please note that this driver is used for both the DraCo Motion and the Toccata.

wavetools.audio

Requires dad_audio.device.

----- Footnotes -----

(1) The latest version of the Delfina software can be found at Petsoff Limited Partnership's WWW page: <http://www.sci.fi/~petsoff>.

(2) maestix.library is available from AmiNet, for example <ftp://ftp.germany.aminet.org/pub/aminet/util/libs/Maestix.lha>.

(3) Richard Körber's WWW page:
<http://www.is-koeln.de/einwohner/shred>.

(4) melodympeg.device and the latest version of this driver can be found at the Kato Development Group's WWW page:
<http://home.pages.de/~kato>.

(5) toccata.library is available from AmiNet, for example <ftp://ftp.germany.aminet.org/pub/aminet/util/libs/toclib12.lha>.

1.19 ahiusr.guide/The Mode Descriptors

The Mode Descriptors

The files in DEVS:AudioModes describes the available audio modes that you can chose from in the audio mode requester. All files located in this drawer will be scanned the first time AHI is used, and added to the internal mode database.

The following modes are available for most drivers:

Mono

Mono output, all sounds will be centered, and no surround sound is possible.

Stereo

Stereo output, but all sounds are either forces to the extreme left or extreme right--centered sounds are not possible, for example. This mode will probably use slightly more CPU power than the "mono" mode. Since all sounds are forced to one of the extreme positions, stereo sounds will play with either the left or the right part muted.

Stereo++

Stereo output with free positioning of all sounds--an instrument can be placed anywhere between the two speakers, for example. Unless the program only uses the extremes when positioning a sound, this mode will eat more CPU cycles than the "stereo" mode.

Fast mono

Fast stereo

Fast stereo++

"Fast" modes take some shortcuts in order to reduce the CPU usage. For 8 bit samples, multiplication tables will be used, which speeds up mixing by magnitudes and still gives the same quality with the exception of volume levels--instead of 256 levels only 32 are available. The disadvantage of multiplication tables is that they require a fair amount of free RAM. For 16 bit samples, the shortcuts are a bit more crude: the volume of each sound will be rounded to a power of 2 before it's played. This means that a 16 bit sound will only be played at volume levels of 100%, 50%, 25%, 12.5% etc. If surround sounds are played, the normal (not "fast") mixing routines will be used. You can use the preferences program (see

Advanced settings

) to disable surround sounds for "fast"

modes.

"Fast" modes are useful if you wish to use as little CPU power as possible, but don't mind spending some memory to reach that goal. They are also very useful when playing 16 bit audio streams--from a sample or MPEG player, for example--since that usually doesn't

involve volume scaling and frequency recalculation. However, if used for playing streams, make sure the mixing frequency is the same as the stream frequency!

HiFi mono

HiFi stereo

HiFi stereo++

"HiFi" modes use much better mixing algorithms than the other modes, using 32 bits internally and linear interpolation. They are also much slower than the other modes. "HiFi" modes turn on master volume with clipping automatically (see Advanced settings).

"HiFi" modes are useful when you're playing music--MIDI songs or modules, for example--and want the best possible quality AHI has to offer.

1.20 ahiusr.guide/The Author

The Author

The author can be reached at the following addresses:

Electronic mail

<martin@blom.org>

Standard mail

Martin Blom
Alsättersgatan 15A:24
SE-584 35 Linköping
SWEDEN

World-Wide Web

<http://martin.blom.org>

The Man

The Myth

The Concept

1.21 ahiusr.guide/The Man

The Man

=====

Martin Blom was born 1974 in a town in Sweden called Jönköping. He had a happy childhood, lots of good friends, and a great family. He did his homework and went to church every Sunday.

But then, one cold, dark Christmas Eve in the year of our Lord 1986, everything went wrong. This was the day when it entered his life. At once, there were fights among the brothers. They all wanted to use it. Martin started to avoid playing with kids that didn't share his passion for it. The school work suffered. Other interests suffered. It was the Commodore 64 home computer, and it would forever change his life.

Today, more than ten years after the tragedy, things are worse than ever. He is studying Computer Science and Engineering at Linköping Institute of Technology, surrounded every day by other computer nerds.

Martin has spent loads of money on computers over the years: Amiga 500, Amiga 4000/040, Commodore 128D, Commodore 64 (in order of appearance), modem, monitors, disks, mice etc. Interesting enough, no sound card. He did, however, build a sound card of his own for the Commodore 64, and he likes to mention that now and then (you see, this was one of the few hardware projects that actually worked!). 4 channels, 8 bit samples. He even wrote a module player for the good old 64. And it had quadrascopes.

1.22 ahiusr.guide/The Myth

The Myth
=====

Some people actually seem to believe that Martin is a good programmer. They couldn't be more wrong. He is lazy, has no patience, he is a slow thinker and he doesn't like anything he has to do.

Martin used to say

- * If you can't write applications, write games.
- * If you can't write games, write demos.
- * If you can't write demos, write utilities.
- * If you can't write utilities, write BBS intros and doors.
- * If you can't do that either, get a modem and start trading.

And guess what? He tried demos. He tried utilities. He tried intros. He wrote a door for /X. And he traded warez.

1.23 ahiusr.guide/The Concept

The Concept

=====

What do you do if you don't have the patience to write applications, if you only write moderate demos, are tired of utilities, hate BBS doors, are totally fed up with playing games and have decided to get legal and stop pirating software? Simple. Try a new concept!

Take a deep breath. Close your eyes. Think of one thing your computer lacks. Think of one of the things that makes your favorite toy feel outdated. Think of something that nobody has (successfully) tried before. Then write the software, and release it as Freeware.

In Martins case, that something was hardware independent audio.

Come on, admit it! It's brilliant. It doesn't matter if you are a good programmer. It doesn't matter if it takes 3 years to get to a half-finished product. It doesn't matter if you give it the most unimaginative name in the world--you can even use a TLA (1). Nobody is going to say your software sucks, because nobody can say he has done better himself. Nobody is going to complain if you're slow on releasing bug fixes and updates, because the software is free. And nobody is going to be angry with you if you stop developing the software--because it sucked in the first place, remember?

This concept won't make you rich, but are rich people really happier?

----- Footnotes -----

(1) Three Letter Acronym

1.24 ahiusr.guide/Acknowledgments

Acknowledgments

The author wish to give special thanks to the following persons (in alphabetical order):

Amiga Translators' Organization
For the catalog translations.

Daniel Arthursson and Johan Nyblom
For making it possible to write the first driver for a sound card, the Wavetools card. I'm sure AHI would not have been accepted as quick as it was without this driver.

Christian Buchner
For the calibrated 14 bit routines for Paula.

MacroSystem Computer GmbH

For lending me a Toccata card.

Steve Krueger and SAS Institute, Inc.
For the compiler and all the updates.

Johan Otterström
For all the help with the Toccata driver.

Jyrki Petsalo and Teemu Suikki
For the Delfina driver, and for supporting AHI in the early days.
And of course, for the sound card!

Pauli Porkka
For active support and promotion of AHI from the beginning.

All the rest
Many, many other have helped me, sent suggestions etc. I owe you
a lot.

And of course, the actual catalog translators: Samuel Aguilera,
Andrija Antonijevic Rúben Alvim, Stéphane Barbaray, Frederico Borges,
Piergiorgio Ghezze, Roger Hågensen, Bernardo Innocenti, Ljubomir
Jankovic, Petteri Kallio, Siniş^1şa Loliæ, Eivind Olsen, Marcin Orłowski,
Thomas Petersen, Pauli Porkka, Vit Sindlar, Martin Sprenger, Sönke
Tesch, Vörös Viktor, Michel Vissers, Ondrej Zima, me, myself and I....

Contributors

Who has contributed to the AHI project?

1.25 ahiusr.guide/Contributors

Contributors
=====

The following people has contributed to the AHI project with code:

Stéphane Barbaray <opty@club-internet.fr>
The MUI version of the preferences program.

Olaf Barthel <olsen@sourcery.han.de>
The Concierto driver.

Thorsten Hansen <hansen_t@informatik.fh-hamburg.de>
The Melody MPEG driver.

Richard Körber <shred@chessy.aworld.de>
The Maestro Pro driver.

Johan Nyblom <nyblom@ludd.luth.se>
The latest Wavetools driver.

Pauli Porkka <pporkka@iki.fi>
The first version of the Toccata driver.

Rüdiger Sopp
The first version of the preferences program. Too bad things
didn't work out as planned.

Teemu Suikki <tsuikki@lut.fi>
The Delfina driver.

Thomas Wenzel <wenzel@unixserv.rz.fh-hannover.de>
The Prelude driver.

Many thanks!

1.26 ahiusr.guide/Concept Index

Concept Index

14 bit calibration
The Drivers

68060 CPU
The Drivers

Acknowledgments
Acknowledgments

AddAudioModes <1>
AddAudioModes

AddAudioModes
The Drivers

Advanced settings
Advanced settings

AHI, overview
Overview

AHI, preferences program
AHI

AHI-Handler
AHI-Handler

AHIpaulaBufferLength variable
The Drivers

AHIpaulaFakeMixFreq variable
The Drivers

AHIpaulaFilterFreq variable
The Drivers

AHIpaulaSampleLimit variable
The Drivers

AHIpaulaSwapChannels variable
The Drivers

AHItoccataIrqSize variable
The Drivers

AHItoccataNoTask variable
The Drivers

AIFC
AHI-Handler

AIFF
AHI-Handler

AmigaVideo
The Drivers

Audio mode database
The Mode Descriptors

Audio mode settings
Mode settings

Audio modes, overview
The Mode Descriptors

AUDIO:
AHI-Handler

Aura
System Files

Author of AHI
The Author

Calibration, 14 bit
The Drivers

Clarity
System Files

Clipping, master volume
Advanced settings

Concierto
System Files

Contributors
Contributors

Copyright
Distribution

CPU usage limit
Advanced settings

CyberGraphX
The Drivers

CyberSound
The Drivers

DAC calibration
The Drivers

Database, audio modes
The Mode Descriptors

Debug level
Advanced settings

Delfina DSP
System Files

Delfina Lite
System Files

Disclaimer
Distribution

Distortion, while recording
AHI-Handler

Distribution
Distribution

Donations
Donations

DraCo Motion
System Files

Drivers
System Files

Echo, enable-disable-fast
Advanced settings

Fame
The Concept

Fast Mono (audio mode)
The Mode Descriptors

Fast Stereo (audio mode)
The Mode Descriptors

Fast Stereo++ (audio mode)
The Mode Descriptors

HiFi Mono (audio mode)
The Mode Descriptors

HiFi Stereo (audio mode)
The Mode Descriptors

HiFi Stereo++ (audio mode)
The Mode Descriptors

IFF-AIFC
AHI-Handler

IFF-AIFF
AHI-Handler

Interpolation, linear
The Mode Descriptors

Jesus Christ
The Man

Legal nonsense
Distribution

License
Distribution

Linear interpolation
The Mode Descriptors

Maestro Pro
System Files

Master volume, clipping
Advanced settings

MC68060 CPU
The Drivers

Melody MPEG
System Files

Mode settings
Mode settings

Modules on a C64
The Man

Mono (audio mode)
The Mode Descriptors

Multiplication tables
The Mode Descriptors

- Nerd, definition of
 - The Man
- Overview
 - Overview
- Paula
 - System Files
- Paula, custom chip
 - Overview
- Picasso96
 - The Drivers
- Preferences
 - AHI
- Preferences, advanced
 - Advanced settings
- Preferences, audio mode
 - Mode settings
- Prelude
 - System Files
- Recursion
 - Concept Index
- Settings
 - AHI
- Settings, advanced
 - Advanced settings
- Settings, audio mode
 - Mode settings
- Software license
 - Distribution
- Sound card drivers
 - System Files
- Starvation
 - Donations
- Stereo (audio mode)
 - The Mode Descriptors
- Stereo++ (audio mode)
 - The Mode Descriptors
- Surround sound, enable-disable
 - Advanced settings

System description
System description

System Files
System Files

Tables, multiplication
The Mode Descriptors

The Author
The Author

The Concept
The Concept

The Drivers
The Drivers

The Man
The Man

The Mode Descriptors
The Mode Descriptors

The Myth
The Myth

Toccata
System Files

VGA screen mode
The Drivers

Wavetools
System Files
