AHI

Hardware independent audio for Amiga
AHI Developer's Guide for AHI release 6.0
Document version 5.3.2.3 (2005-02-02)

Martin 'Leviticus' Blom

1 Overview

This document was written in order to make it easier for developers to understand and use AHI in their own productions, and write Software That Works(TM).

`ahi.device` has two different API's; one library-like function interface (low-level), and one "normal" device interface (high-level). Each of them serves different purposes. The low-level interface is targeting music players, games and real-time applications. The high-level interface is targeting applications that just want to have a sample played, play audio streams or record samples as easily as possible.

As with everything else, it is important that you chose the right tool for the job—you'll only get frustrated otherwise.

Not everything about AHI is documented here; for more information, see AHI *User's Guide* and the autodocs.

2 Distribution

Ahi is available under a dual license. The device itself is under the "GNU Library General Public License" (see [LGPL], page 29), while the utility programs and the `AUDIO:` device is covered by the "GNU General Public License" (see [GPL], page 23).

If you use this software in a commercial or shareware product, please consider giving the author (see Chapter 3 [The Author], page 5)—and preferably each one of the contributors too (see Ahi *User's Guide*)—an original or registered copy or sample of your work. Should you want to distribute the Ahi software with your own product, there is really nothing to consider, right?

3 The Author

The author can be reached at the following addresses:

Electronic mail
`martin@blom.org`

Standard mail
Martin Blom
Luftvärnsgatan 42
SE-587 34 Linköping
Sweden

World-Wide Web
`http://martin.blom.org`

4 Definitions

Following are some general definitions of terms that are used in this document.

Sample
: A sample is one binary number, representing the amplitude at a fixed point in time. A sample is often stored as an 8 bit signed integer, a 16 bit signed integer, a 32 bit floating point number etc. AHI only supports integers.

Sample frame
: In mono environments, a sample frame is the same as a sample. In stereo environments, a sample frame is a tuple of two samples. The first member is for the left channel, the second for the right.

Sound
: Many sample frames stored in sequence as an array can be called a sound. A sound is, however, not limited to being formed by samples, it can also be parameters to an analog synth or a MIDI instrument, or be white noise. AHI only supports sounds formed by samples.

5 Function Interface

The device has, in addition to the usual I/O request protocol, a set of functions that allows the programmer to gain full control (at least as much as possible with device independence) over the audio hardware. The advantages are low overhead and much more advanced control over the playing sounds. The disadvantages are greater complexity and only one user per sound card.

If you want to play music or sound effects for a game, record in high quality or want to do realtime effects, this is the API to use.

5.1 Guidelines

5.1.1 Follow The Rules

It's really simple. If I tell you to check return values, check sample types when recording, not to trash d2-d7/a2-a6 in hooks, or not to call `AHI_ControlAudio()` with the `AHIC_Play` tag from interrupts or hooks, you do as you are told.

5.1.2 The Library Base

The `AHIBase` structure is private, so are the sub libraries' library base structures. Don't try to be clever.

5.1.3 The Audio Database

The implementation of the database is private, and may change any time. `ahi.device` provides functions access the information in the database (`AHI_NextAudioID()`, `AHI_GetAudioAttrsA()` and `AHI_BestAudioIDA()`).

5.1.4 User Hooks

All user hooks must follow normal register conventions, which means that d2-d7 and a2-a6 must be preserved. They may be called from an interrupt, but you cannot count on that; it can be your own process or another process. Don't assume the system is in single-thread mode. Never spend much time in a hook, get the work done as quick as possible and then return.

5.1.5 Function Calls From Other Tasks, Interrupts Or User Hooks

The `AHIAudioCtrl` structure may not be shared with other tasks/threads. The task that called `AHI_AllocAudioA()` must do all other calls too (except those callable from interrupts).

Only calls specifically said to be callable from interrupts may be called from user hooks or interrupts. Note that `AHI_ControlAudioA()` has some tags that must not be present when called from an interrupt.

5.1.6 Multitasking

Most audio drivers need multitasking to be turned on to function properly. Don't turn it off while using the device.

5.2 Opening And Closing `ahi.device` For Low-level Access

Not too hard. Just open `ahi.device` unit `AHI_NO_UNIT` and initialize `AHIBase`. After that you can access all the functions of the device just as if you had opened a standard shared library.

5.2.1 Assembler

For the assembler programmer there are two handy macros: `OPENAHI` and `CLOSEAHI`. Here is a small example how to use them:

```
        OPENAHI 4                  ;Open at least version 4.
        lea     _AHIBase(pc),a0
        move.l  d0,(a0)
        beq     error

; AHI's functions can now be called as normal library functions:
        move.l  _AHIBase(pc),a6
        moveq   #AHI_INVALID_ID,d0
        jsr     _LVOAHI_NextAudioID(a6)

error:
        CLOSEAHI
        rts
```

Note that you **have** to execute the `CLOSEAHI` macro even if `OPENAHI` failed!

5.2.2 C

For the C programmer, here is how it should be done:

```
struct Library    *AHIBase;
struct MsgPort    *AHImp=NULL;
struct AHIRequest *AHIio=NULL;
BYTE               AHIDevice=-1;

if(AHImp = CreateMsgPort())
{
  if(AHIio = (struct AHIRequest *) CreateIORequest(
      AHImp, sizeof(struct AHIRequest)))
  {
    AHIio->ahir_Version = 4;  /* Open at least version 4. */
    if(!(AHIDevice = OpenDevice(AHINAME, AHI_NO_UNIT,
        (struct IORequest *) AHIio, NULL)))
    {
      AHIBase = (struct Library *) AHIio->ahir_Std.io_Device;

// AHI's functions can now be called as normal library functions:
      AHI_NextAudioID(AHI_INVALID_ID);

      CloseDevice((struct IORequest *) AHIio);
      AHIDevice = -1;
    }
    DeleteIORequest((struct IORequest *) AHIio);
```

```
        AHIio = NULL;
      }
      DeleteMsgPort(AHImp);
      AHImp = NULL;
    }
```

5.3 Obtaining The Hardware

If you wish to call any other function than

- `AHI_AllocAudioRequestA()`
- `AHI_AudioRequestA()`
- `AHI_BestAudioIDA()`
- `AHI_FreeAudioRequest()`
- `AHI_GetAudioAttrsA()`
- `AHI_NextAudioID()`
- `AHI_SampleFrameSize()`

. . .you have to allocate the actual sound hardware. This is done with `AHI_AllocAudioA()`. `AHI_AllocAudioA()` returns an `AHIAudioCtrl` structure, or `NULL` if the hardware could not be allocated. The `AHIAudioCtrl` structure has only one public field, `ahiac_UserData`. This is unused by AHI and you may store anything you like here.

If `AHI_AllocAudioA()` fails it is important that you handle the situation gracefully.

When you are finished playing or recording, call `AHI_FreeAudio()` to deallocate the hardware and other resources allocated by `AHI_AllocAudioA()`. `AHI_FreeAudio()` also deallocates all loaded sounds (see Section 5.4 [Declaring Sounds], page 12).

5.3.1 `AHI_AllocAudioA()` Tags

`AHI_AllocAudioA()` takes several tags as input.

`AHIA_AudioID`
: This is the audio mode to be used. You must not use any hardcoded values other than `AHI_DEFAULT_ID`, which is the user's default fallback ID. In most cases you should ask the user for an ID code (with `AHI_AudioRequestA()`) and then store the value in your settings file.

`AHIA_MixFreq`
: This is the mixing frequency to be used. The actual frequency will be rounded to the nearest frequency supported by the sound hardware. To find the actual frequency, use `AHI_GetAudioAttrsA()`. If omitted or `AHI_DEFAULT_FREQ`, the user's preferred fallback frequency will be used. In most cases you should ask the user for a frequency (with `AHI_AudioRequestA()`) and then store the value in your settings file.

`AHIA_Channels`
: All sounds are played on a *channel*, and this tag selects how many you wish to use. In general it takes more CPU power the more channels you use and the volume gets lower and lower.

`AHIA_Sounds`
: You must tell AHI how many different sounds you are going to play. See Section 5.4 [Declaring Sounds], page 12 for more information.

AHIA_SoundFunc

With this tag you tell AHI to call a hook when a sound has been started. It works just like Paula's audio interrupts. The hook receives an AHISoundMessage structure as message. AHISoundMessage->ahism_Channel indicates which channel the sound that caused the hook to be called is played on.

AHIA_PlayerFunc

If you are going to play a musical score, you should use this "interrupt" source instead of VBLANK or CIA timers in order to get the best result with all audio drivers. If you cannot use this, you must not use any "non-realtime" modes (see AHI_GetAudioAttrsA() in the autodocs, the AHIDB_Realtime tag).

AHIA_PlayerFreq

If non-zero, it enables timing and specifies how many times per second PlayerFunc will be called. This must be specified if AHIA_PlayerFunc is! It is suggested that you keep the frequency below 100-200 Hz. Since the frequency is a fixpoint number AHIA_PlayerFreq should be less than 13107200 (that's 200 Hz).

AHIA_MinPlayerFreq

The minimum frequency (AHIA_PlayerFreq) you will use. You should always supply this if you are using the device's interrupt feature!

AHIA_MaxPlayerFreq

The maximum frequency (AHIA_PlayerFreq) you will use. You should always supply this if you are using the device's interrupt feature!

AHIA_RecordFunc

This hook will be called regularly when sampling is turned on (see AHI_ControlAudioA()). It is important that you always check the format of the sampled data, and ignore it if you can't parse it. Since this hook may be called from an interrupt, it is not legal to directly Write() the buffer to disk. To record directly to harddisk you have to copy the samples to another buffer and signal a process to save it. To find out the required size of the buffer, see AHI_GetAudioAttrsA() in the autodocs, the AHIDB_MaxRecordSamples tag.

AHIA_UserData

Can be used to initialize the ahiac_UserData field. You do not have to use this tag to change ahiac_UserData, you may write to it directly.

AHIA_AntiClickSamples

New for version 6, this tag specifies how many sample frames a sound may be delayed when started in order to reduce clicking. In practice, the currently playing sound will continue until a zero-crossing is found or AHIA_AntiClickSamples samples have been processed. After that, the new sound will be started.

The default for this tag can be set by the user in the preferences program. Set it to 0 to disable this feature.

5.4 Declaring Sounds

Before you can play a sample array, you must AHI_LoadSound() it. Why? Because if AHI knows what kind of sounds that will be played later, tables and stuff can be set up in advance. Some drivers may even upload the samples to the sound cards local RAM and play all samples from there, drastically reducing CPU and bus load.

You should AHI_LoadSound() the most important sounds first, since the sound cards RAM may not be large enough to hold all your sounds.

AHI_LoadSound() also associates each sound or sample array with a number, which is later used to refer to that particular sound.

There are 2 types of sounds, namely AHIST_SAMPLE and AHIST_DYNAMICSAMPLE.

AHIST_SAMPLE
: This is used for static samples. Most sounds that will be played are of this type. Once the samples have been "loaded", you may not alter the memory where the samples are located. You may, however, read from it.

AHIST_DYNAMICSAMPLE
: If you wish to play samples that you calculate in realtime, or load in portions from disk, you must use this type. These samples will never be uploaded to a sound cards local RAM, but always played from the normal memory. There is a catch, however. Because of the fact that the sound is mixed in chunks, you must have a certain number of samples in memory before you start a sound of this type. To calculate the size of the buffer (in samples), use the following formula:

$$size = samples * Fs / Fm$$

where samples is the value returned from AHI_GetAudioAttrsA() when called with the AHIDB_MaxPlaySamples tag, Fs is the highest frequency the sound will be played at and Fm is the actual mixing frequency (AHI_ControlAudioA()/AHIC_MixFreq_Query).

The samples can be in one of seven different formats, named AHIST_M8S, AHIST_S8S, AHIST_M16S, AHIST_S16S, AHIST_M32S, AHIST_S32S and AHIST_L7_1

AHIST_M8S
: This is an 8 bit mono sound. Each sample frame is just one signed byte.

AHIST_S8S
: This is an 8 bit stereo sound. Each sample frame is one signed byte representing the left channel, followed by another one for the right channel.

AHIST_M16S
: This is a 16 bit mono sound. Each sample frame is just one signed 16 bit word, in big endian/network order format (most significant byte first).

AHIST_S16S
: This is a 16 bit stereo sound. Each sample frame is one signed 16 bit word, in big endian/network order format (most significant byte first) representing the left channel, followed by another one for the right channel.

AHIST_M32S
: This is a 32 bit mono sound. Each sample frame is just one signed 32 bit word, in big endian/network order format (most significant byte first). Note that only the 24 most significant bits are guaranteed to be processed correctly! Support for this sample format was added in V6.

AHIST_S32S
: This is a 32 bit stereo sound. Each sample frame is one signed 32 bit word, in big endian/network order format (most significant byte first) representing the left channel, followed by another one for the right channel. Note that only the 24 most significant bits are guaranteed to be processed correctly! Support for this sample format was added in V6.

AHIST_L7_1
: This is a 32 bit 7.1 sound. **It will currently only work with 7.1 audio modes!** Each sample frame is one signed 32 bit word, in big endian/network order format (most

significant byte first) representing the left front channel, followed by six other words for the right front, left back, right back, left surround, right surround, front center and the LFE channel. Note that only the 24 most significant bits are guaranteed to be processed correctly! "Support" for this sample format was added in V6.

If you know that you won't use a sound anymore, call `AHI_UnloadSound()`. `AHI_FreeAudio()` will also do that for you for any sounds left when called.

There is no need to place a sample array in *Chip memory*, but it **must not** be swapped out! Allocate your sample memory with the `MEMF_PUBLIC` flag set. If you wish to have your samples in virtual memory, you have to write a double-buffer routine that copies a chunk of memory to a `MEMF_PUBLIC` buffer. The *SoundFunc* should signal a task to do the transfer, since it may run in supervisor mode (see `AHI_AllocAudioA()`).

5.5 Making Noise

After you have allocated the sound hardware and declared all your sounds, you're ready to start playback. This is done with a call to `AHI_ControlAudioA()`, with the `AHIC_Play` tag set to `TRUE`. When this function returns the *PlayerFunc* (see `AHI_AllocAudioA()`) is active, and the audio driver is feeding silence to the sound hardware.

5.5.1 Playing A Sound

All you have to do now is to set the desired sound, it's frequency and volume. This is done with `AHI_SetSound()`, `AHI_SetFreq()` and `AHI_SetVol()`. Make sure the `AHISF_IMM` flag is set for all these function's *flag* argument. And don't try to modify a channel that is out of range! If you have allocated 4 channels you may only modify channels 0-3.

The sound will not start until both `AHI_SetSound()` and `AHI_SetFreq()` have been called. The sound will play even if `AHI_SetVol()` was not called, but it will play completely silent. If you wish to temporary stop a sound, set its frequency to 0. When you change the frequency again, the sound will continue where it was.

The actual beginning of sound might be delayed slightly, depending on the value of the `AHIA_AntiClickSamples` tag passed to `AHI_AllocAudioA()`. Should you wish to override this, set the `AHISF_NODELAY` in addition to `AHISF_IMM`.

When the sound has been started it will play to the end and then repeat. In order to play a one-shot sound you have use the `AHI_PlayA()` function, or install a sound interrupt using the `AHIA_SoundFunc` tag with `AHI_AllocAudioA()`. For more information about using sound interrupts, see below.

A little note regarding `AHI_SetSound()`: *Offset* is the first sample frame that will be played, both when playing backwards and forwards. This means that if you call `AHI_SetSound()` with *offset* 0 and *length* 4, sample fram 0, 1, 2 and 3 will be played. If you call `AHI_SetSound()` with *offset* 3 and *length* −4, sample frame 3, 2, 1 and 0 will be played.

Also note that playing very short sounds will be very CPU intensive, since there are many tasks that must be done each time a sound has reached its end (like starting the next one, calling the *SoundFunc*, etc.). Therefore, it is recommended that you "unroll" short sounds a couple of times before you play them. How many times you should unroll? Well, it depends on the situation, of course, but try making the sound a thousand samples long if you can. Naturally, if you need your *SoundFunc* to be called, you cannot unroll.

5.5.2 Playing One-shot Sounds And Advanced Loops

In version 4, some changes have been made since earlier releases. One-shot sounds and sounds with only one loop segment can now be played without using sample interrupts. This is possible because one of the restrictions regarding the `AHISF_IMM` flag has been removed.

The `AHISF_IMM` flag determines if `AHI_SetSound()`, `AHI_SetFreq()` and `AHI_SetVol()` should take effect immediately or when the current sound has reached its end. The rules for this flags are:

- If used inside a sample interrupt (*SoundFunc*): Must be cleared.
- If used inside a player interrupt (*PlayerFunc*): May be set or cleared.
- If used elsewhere: Must be set.

What does this mean? It means that if all you want to do is to play a one-shot sound from inside a *PlayerFunc*, you can do that by first calling `AHI_SetSound()`, `AHI_SetFreq()` and `AHI_SetVol()` with `AHISF_IMM` set, and then use `AHI_SetSound(ch, AHI_NOSOUND, 0, 0, actrl, 0L)` to stop the sound when it has reached the end. You can also set one loop segment this way.

`AHI_PlayA()` was added in AHI version 4, and combines `AHI_SetSound()`, `AHI_SetFreq()` and `AHI_SetVol()` into one tag-based function. It also allows you to set one loop and play one-shot sounds.

To play a sound with more than one loop segment or ping-pong looping, a sample interrupt needs to be used. AHI's *SoundFunc* works like Paula's interrupts and is very easy to use.

The *SoundFunc* hook will be called with an `AHIAudioCtrl` structure as object and an `AHISoundMessage` structure as message. `ahism_Channel` indicates which channel caused the hook to be called.

An example *SoundFunc* which handles the repeat part of an instrument can look like this (SAS/C code):

```
__asm __saveds ULONG SoundFunc(register __a0 struct Hook *hook,
    register __a2 struct AHIAudioCtrl *actrl,
    register __a1 struct AHISoundMessage *chan)
{
  if(ChannelDatas[chan->ahism_Channel].Length)
    AHI_SetSound(chan->ahism_Channel, 0,
        (ULONG) ChannelDatas[chan->ahism_Channel].Address,
        ChannelDatas[chan->ahism_Channel].Length,
        actrl, NULL);
  else
    AHI_SetSound(chan->ahism_Channel, AHI_NOSOUND,
        NULL, NULL, actrl, NULL);
  return NULL;
}
```

This example is from an old version of the AHI NotePlayer for *DeliTracker 2*. `ChannelDatas` is an array where the start and length of the repeat part is stored. Here, a repeat length of zero indicates a one-shot sound. Note that this particular example only uses one sound (0). For applications using multiple sounds, the sound number would have to be stored in the array as well.

Once again, note that the `AHISF_IMM` flag should **never** be set in a *SoundFunc* hook!

5.5.3 Tricks With The Volume

Starting with V4, `AHI_SetVol()` can take both negative volume and pan parameters. If you set the volume to a negative value, the sample will, if the audio mode supports it, invert each sample before playing. If pan is negative, the sample will be encoded to go to the surround speakers.

6 Device Interface

The I/O request protocol makes it very easy to play audio streams, sounds from disk and non time-critical sound effects in a multitasking friendly way. Recoding is just as easy, on behalf of quality. Several programs can play sounds at the same time, and even record at the same time if your hardware is full duplex.

If you want to write a sample player, play (warning?) sounds in your applications, play an audio stream from a CD via the SCSI/IDE bus, write a voice command utility etc., this is the API to use.

Note that while all the low-level functions (see Chapter 5 [Function Interface], page 9) count lengths and offsets in sample frames, the device interface—like all Amiga devices—uses bytes.

6.1 Opening And Closing `ahi.device` For High-level Access

Four primary steps are required to open ahi.device:

- Create a message port using `CreateMsgPort()`. Reply messages from the device must be directed to a message port.
- Create an extended I/O request structure of type `AHIRequest` using `CreateIORequest()`. `CreateIORequest()` will initialize the I/O request to point to your reply port.
- Specify which version of the device you need. The lowest supported version is 4. Version 1 and 3 are obsolete, and version 2 only has the low-level API.
- Open `ahi.device` unit `AHI_DEFAULT_UNIT` or any other unit the user has specified with, for example, a *UNIT* tooltype. Call `OpenDevice()`, passing the I/O request.

Each `OpenDevice()` must eventually be matched by a call to `CloseDevice()`. When the last close is performed, the device will deallocate all resources.

All I/O requests must be completed before `CloseDevice()`. Abort any pending requests with `AbortIO()`.

Example:

```
struct MsgPort    *AHImp      = NULL;
struct AHIRequest *AHIio      = NULL;
BYTE               AHIDevice  = -1;
UBYTE              unit       = AHI_DEFAULT_UNIT;

/* Check if user wants another unit here... */

if(AHImp = CreateMsgPort())
{
  if(AHIio = (struct AHIRequest *)
      CreateIORequest(AHImp, sizeof(struct AHIRequest)))
  {
    AHIio->ahir_Version = 4;
    if(!(AHIDevice = OpenDevice(AHINAME, unit,
        (struct IORequest *) AHIio, NULL)))
    {


      /* Send commands to the device here... */
```

```
        if(! CheckIO((struct IORequest *) AHIio))
        {
          AbortIO((struct IORequest *) AHIio);
        }

        WaitIO((struct IORequest *) AHIio);

        CloseDevice((struct IORequest *) AHIio);
        AHIDevice = -1;
      }
      DeleteIORequest((struct IORequest *) AHIio);
      AHIio = NULL;
    }
    DeleteMsgPort(AHImp);
    AHImp = NULL;
  }
```

6.2 Reading From The Device

You read from `ahi.device` by passing an `AHIRequest` to the device with `CMD_READ` set in `io_Command`, the number of bytes to be read set in `io_Length`, the address of the read buffer set in `io_Data`, the desired sample format set in `ahir_Type` and the desired sample frequency set in `ahir_Frequency`. The first read command in a sequence should also have `io_Offset` set to 0. `io_Length` must be an even multiple of the sample frame size.

6.2.1 Double Buffering

To do double buffering, just fill the first buffer with `DoIO()` and `io_Offset` set to 0, then start filling the second buffer with `SendIO()` using the same I/O request (but don't clear `io_Offset`!). After you have processed the first buffer, wait until the I/O request is finished and start over with `SendIO()` on the first buffer.

6.2.2 Distortion

The samples will automatically be converted to the sample format set in `ahir_Type` and to the sample frequency set in `ahir_Frequency`. Because it is quite unlikely that you ask for the same sample frequency the user has chosen in the preference program, chances that the quality is lower than expected are pretty high. The worst problem is probably the anti-aliasing filter before the A/D converter. If the user has selected a higher sampling/mixing frequency than you request, the signal will be distorted according to the Nyquist sampling theorem. If, on the other hand, the user has selected a lower sampling/mixing frequency than you request, the signal will not be distorted but rather bandlimited more than necessary.

6.3 Writing To The Device

You write to the device by passing an `AHIRequest` to the device with `CMD_WRITE` set in `io_Command`, the precedence in `io_Message.mn_Node.ln_Pri`, the number of bytes to be written in `io_Length`, the address of the write buffer set in `io_Data`, the sample format set in `ahir_Type`, the desired sample frequency set in `ahir_Frequency`, the desired volume set in `ahir_Volume` and the desired stereo position set in `ahir_Position`. Unless you are doing double buffering, `ahir_Link` should be set to `NULL`. `io_Length` must be an even multiple of the sample frame size.

6.3.1 Double Buffering

To do double buffering, you need two I/O requests. Create the second one by making a copy of the request you used in `OpenDevice()`. Start the first with `SendIO()`. Set `ahir_Link` in the second request to the address of the first request, and `SendIO()` it. Wait on the first, fill the first buffer again and repeat, this time with `ahir_Link` of the first buffer set to the address of the second I/O request.

6.3.2 Distortion

The problems with aliasing are present but not as obvious as with reading. Just make sure your source data is bandlimited correctly, and do not play samples at a lower frequency than they were recorded.

6.3.3 Playing multiple sounds at the same time

If you want to play several sounds at the same time, just make a new copy of the I/O request you used in `OpenDevice()`, and `CMD_WRITE` it. The user has set the number of channels available in the preference tool, and if too many requests are sent to the device the one with lowest precedence will be muted. When a request is finished, the muted request with the highest precedence will be played. Note that all muted requests continue to play silently, so the programmer will not have to worry if there are enough channels or not.

6.3.4 Suggested precedences

The precedences to use depend on what kind of sound you are playing. The recommended precedences are the same as for `audio.device`, listed in *AMIGA ROM Kernel Reference manual – Devices*. Reprinted without permission. So sue me.

```
Precedences  | Type of sound
-------------+----------------------------------------------------------
   127       |  Unstoppable.  Sounds first allocated at lower
             | precedencies, then set to this highest level.
   90 - 100  |  Emergencies.  Alert, urgent situation that requires
             | immediate action.
   80 - 90   |  Annunciators.  Attention, bell (CTRL-G).
   75        |  Speech.  Synthesized or recorded speech
             | (narrator.device).
   50 - 70   |  Sonic cues.  Sounds that provide information that is not
             | provided by graphics.  Only the beginning of of each sound
             | should be at this level; the rest should ne set to sound
             | effects level.
  -50 - 50   |  Music program.  Musical notes in a music-oriented program.
             | The higher levels should be used for the attack portions
             | of each note.
  -70 - -50  |  Sound effects.  Sounds used in conjunction with graphics.
             | More important sounds should use higher levels.
  -100 - -80 |  Background.  Theme music and restartable background sounds.
  -128       |  Silence.  Lowest level (freeing the channel completely is
```

```
                    | preferred).
```

Right. As you can see, some things do not apply to `ahi.device`. First, there is no way to change the precedence of a playing sound, so the precedences should be set from the beginning. Second, it is not recommended to use the device interface to play music. However, playing an audio stream from CD or disk comes very close. Third, there are no channels to free in Ahi since they are dynamically allocated by the device.

7 Data Types And Structures

In this chapter some of the data types and structures used will be explained. For more information, please consult the autodocs and the include files.

7.1 Data Types

7.1.1 `Fixed`

`Fixed` is a signed long integer. It is used to represent decimal numbers without using floating point arithmetics. The decimal point is assumed to be in the middle of the 32 bit integer, thus giving 16 bits for the integer part of the number and 16 bits for the fraction. The largest number that can be stored in a `Fixed` is +32767.999984741, and the lowest number is −32768.

Example:

```
Decimal | Fixed
--------+----------
 1.0    | 0x00010000
 0.5    | 0x00008000
 0.25   | 0x00004000
 0      | 0x00000000
-0.25   | 0xffffc000
-0.5    | 0xffff8000
-1.0    | 0xffff0000
```

7.1.2 `sposition`

`sposition` (stereo position) is a `Fixed`, and is used to represent the stereo position of a sound. 0 is far left, 0.5 is center and 1.0 is far right.

7.2 Structures

7.2.1 `AHIUnitPrefs` And `AHIGlobalPrefs`

These structures are used in the `AHIU` and `AHIG` chunks, respective, which are part of the settings file (‘`ENV:Sys/ahi.prefs`’), The file is read by AHI on each call to `OpenDevice()`, just before the audio hardware is allocated.

`AHIUnitPrefs` specifies the audio mode and its parameters to use for each device unit (currently 0-3 and `AHI_NO_UNIT`; unit 0 is also called `AHI_DEFAULT_UNIT`).

`AHIGlobalPrefs` contains some global options that can be used to gain speed on slow CPUs, the global debug level and a protection against CPU overload. The debug level specifies which of the functions in AHI should print debugging information to the serial port (the output can be redirected to a console window or a file with tools like *Sushi*[1]).

[1] Available from AmiNet, for example `ftp://ftp.germany.aminet.org/pub/aminet/dev/debug/Sushi.lha`.

GNU GENERAL PUBLIC LICENSE

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

   You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

   b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

   c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

   In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

   a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

   b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

   If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

   It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

   Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.
10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "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 PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM 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 PROGRAM (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 PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the program's name and an idea of what it does.
Copyright (C) 19yy  name of author

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License along
with this program; if not, write to the Free Software Foundation, Inc.,
59 Temple Place, Suite 330, Boston, MA 02111-1307, USA.
```

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

```
Gnomovision version 69, Copyright (C) 19yy name of author
Gnomovision comes with ABSOLUTELY NO WARRANTY; for details
type 'show w'.  This is free software, and you are welcome
to redistribute it under certain conditions; type 'show c'
for details.
```

The hypothetical commands '`show w`' and '`show c`' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than '`show w`' and '`show c`'; they could even be mouse-clicks or menu items—whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright
interest in the program 'Gnomovision'
(which makes passes at compilers) written
by James Hacker.

signature of Ty Coon, 1 April 1989
Ty Coon, President of Vice
```

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

GNU LIBRARY GENERAL PUBLIC LICENSE

Version 2, June 1991

[This is the first released version of the library GPL. It is
numbered 2 because it goes with version 2 of the ordinary GPL.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software—to make sure the software is free for all its users.

This license, the Library General Public License, applies to some specially designated Free Software Foundation software, and to any other libraries whose authors decide to use it. You can use it for your libraries, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library, or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link a program with the library, you must provide complete object files to the recipients so that they can relink them with the library, after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

Our method of protecting your rights has two steps: (1) copyright the library, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the library.

Also, for each distributor's protection, we want to make certain that everyone understands that there is no warranty for this free library. If the library is modified by someone else and passed on, we want its recipients to know that what they have is not the original version, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that companies distributing free software will individually obtain patent licenses, thus in effect transforming the program into proprietary software. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License, which was designed for utility programs. This license, the GNU Library General Public License, applies to certain designated libraries. This license is quite different from the ordinary one; be sure to read it in full, and don't assume that anything in it is the same as in the ordinary license.

The reason we have a separate public license for some libraries is that they blur the distinction we usually make between modifying or adding to a program and simply using it. Linking a program with a library, without changing the library, is in some sense simply using the library, and is analogous to running a utility program or application program. However, in a textual and legal sense, the linked executable is a combined work, a derivative of the original library, and the ordinary General Public License treats it as such.

Because of this blurred distinction, using the ordinary General Public License for libraries did not effectively promote software sharing, because most developers did not use the libraries. We concluded that weaker conditions might promote sharing better.

However, unrestricted linking of non-free programs would deprive the users of those programs of all benefit from the free status of the libraries themselves. This Library General Public License is intended to permit developers of non-free programs to use free libraries, while preserving your freedom as a user of such programs to change the free libraries that are incorporated in them. (We have not seen how to achieve this as regards changes in header files, but we have achieved it as regards changes in the actual functions of the Library.) The hope is that this will lead to faster development of free libraries.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, while the latter only works together with the library.

Note that it is possible for a library to be covered by the ordinary General Public License rather than by this special one.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Library General Public License (also called "this License"). Each licensee is addressed as "you".

   A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

   The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

   "Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

   Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the

notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

   a. The modified work must itself be a software library.

   b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

   c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

   d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

      (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

   These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

   Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

   In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

   Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

   This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which

must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

   However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

   When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

   If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

   Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also compile or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

   You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

   a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
   b. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
   c. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

   d. Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

   For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

   It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:
   a. Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
   b. Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.
8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.
9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.
10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.
11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

    If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.
13. The Free Software Foundation may publish revised and/or new versions of the Library General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

    Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.
14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "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 LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.
16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY 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 LIBRARY (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 LIBRARY TO OPERATE WITH

ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
one line to give the library's name and an idea of what it does.
Copyright (C) year  name of author

This library is free software; you can redistribute it and/or modify it
under the terms of the GNU Library General Public License as published
by the Free Software Foundation; either version 2 of the License, or (at
your option) any later version.

This library is distributed in the hope that it will be useful, but
WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
Library General Public License for more details.

You should have received a copy of the GNU Library General Public
License along with this library; if not, write to the Free Software
Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

```
Yoyodyne, Inc., hereby disclaims all copyright interest in the library
'Frob' (a library for tweaking knobs) written by James Random Hacker.

signature of Ty Coon, 1 April 1990
Ty Coon, President of Vice
```

That's all there is to it!

Concept Index

Data Type Index

Function Index

Variable Index

A

C

I

L

Table of Contents