The Sound-Trecker: Documentation

05/04/92

Frank Seide • Koolbarg 39d • D-2000 Hamburg 74 • Germany

Table of contents

Table of figures

Many thanks to Olaf Höhmann, who had the idea to write the Sound-Trecker and gave me the source code of an Amiga SoundTracker routine. Thanks to Marc Holstein for his extensive tests and critical remarks, and to Martin Dencker and Martin Willmann of M&M-Marketing, Hamburg.

The Sound-Trecker: Documentation

1 Introduction

The `Sound-Trecker V1.0` is a program to play songs that are stored in a special, compact format. This format was introduced by the program “SoundTracker” written for computers of the Commodore-Amiga family. The music has high output quality, although it is not just digitized, but stored in a compact way as notes. This approach makes it possible to obtain moderate file sizes (between 50 and 300 kBytes) even for songs which are several minutes long. These music files are commonly called “Modules” or “.MOD files”; let's call them just “Soundtracks” in this text. During the last years many high quality songs in “.MOD” format have been created. Usually they are public domain and available through mail boxes, for example.

Amongst several Amiga versions, implementations of “.MOD” file players for Atari ST, PC/AT-compatibles and Acorn Archimedes computers are available. Finally, the Sound-Trecker makes the great amount of sound tracks accessible to the Macintosh users, supposed a MC68020 processor (or higher) and System 6.0.7 (or later) is available. The program is fully multitasking compatible and capable of playing the music in the background. Depending on CPU speed and the selected sound quality, between 76% (Mac LC in best quality) and 10% (Mac Quadra with cache enabled) of the CPU performance is needed.

The Sound-Trecker is a shareware product, which means that the program may be copied freely, and a small amount of money has to be paid to use the program. You may only use the program conforming with the licence agreement contained in an extra file. You may copy the program only together with the licence agreement, the documentation and the included source codes.

In chapter 2 some hints are given about how to transfer sound tracks (“`.MOD`” files) to the Mac. Chapters 3, 4, and 5 explain how to use of Sound-Trecker. Chapter 6 describe some technical details about how the Sound-Trecker works and about the “`.MOD`” file structure. Finally we discuss how to utilize the Sound-Trecker routines to play sound tracks from your own programs.

Compatibility Note

The Sound-Trecker should be compatible with all software that was written conforming with the rules given in “Inside Macintosh”. Problems may arise, when the Sound-Trecker is used together with older software which uses the “Sound Driver” instead of the new “Sound Manager”. As soon as a program calls an old Sound Driver routine, the Sound Manager switches into a compatibility mode and stops *all* sound output generated via Sound Manager calls, including the Sound-Trecker's. When you quit the old program, the Sound Manager immediately restarts sound production. Especially old versions of the shareware program “Sound Master” may cause the Finder to issue a Sound Driver call, and you can't quit the Finder!

2 Transferring Sound Tracks to Your Macintosh

Music files in "`.MOD`" format ("sound tracks") are available by different means, they are, however, still (05/04/92) unusual on the Macintosh. This chapter discusses some ways to transfer sound tracks to your Macintosh.

2.1 Mail Boxes

Many mail boxes contain a "`.MOD`" file department, which is usually located in the Amiga section. After downloading and unpacking, the music files can be played directly by the Sound-Trecker.

2.2 Transferring using PC disks

If a "`.MOD`" file is located on an accessible computer, the easiest way of transferring is to copy the file to a PC compatible disk (720 kBytes or 1.4 MBytes). These disks are used by all PC/AT-compatible as well as all Atari ST computers; special conversion programs are not required. The Amiga disk format is different. There are, however, programs that enable writing and reading of PC disks, for example the commercial program "Dos-2-Dos" or the filing systems "Cross-Dos" and "MSH" (shareware).

As soon as the "`.MOD`" file is available on a PC disk, the Macintosh can read it from this disk. A file conversion program to read PC disks should have been shipped with your system software. Alternatively you can use the system extension "AccessPC" by Insignia Solutions.

2.3 Adapting the File Type

After downloading respectively copying via PC disks, the files usually have the file type `'TEXT'` or `'????'`. You should not try to start such a "fresh" song of music by double clicking (because `TeachText` would try to display the music data as an ASCII text). You should rather start the Sound-Trecker and open the song via the "Open…" menu command (as discussed in chapter 3). The file type will then be adapted automatically..

If you use "AccessPC" for reading PC disks, you may define an automatic type conversion in the "AccessPC" control panel. You just have to define the MS-DOS extension "`.MOD`", and assign it the file type `'STrk'` of the application "`The Sound-Trecker V1.0`".

The Sound-Trecker: Documentation

3 Basic Functions

This chapter explains the basic functions of the Sound-Trecker, such as loading and playing songs, and using the control buttons.

3.1 Starting the Sound-Trecker

The Sound-Trecker can be started be double clicking on the program icon (), just like any other Macintosh application. The symbol is a tractor (German “Trecker”) with notes coming out of its exhaust pipe. Alternatively the Sound-Trecker may be startet by selecting one or more sound tracks in the Finder, and opening them by double clicking. Under System 7 sound tracks may be dragged and dropped upon the Sound-Trecker icon. The Sound-Trecker may also be startetd by opening a configuration file (see chapter 5).

3.2 Screen Display

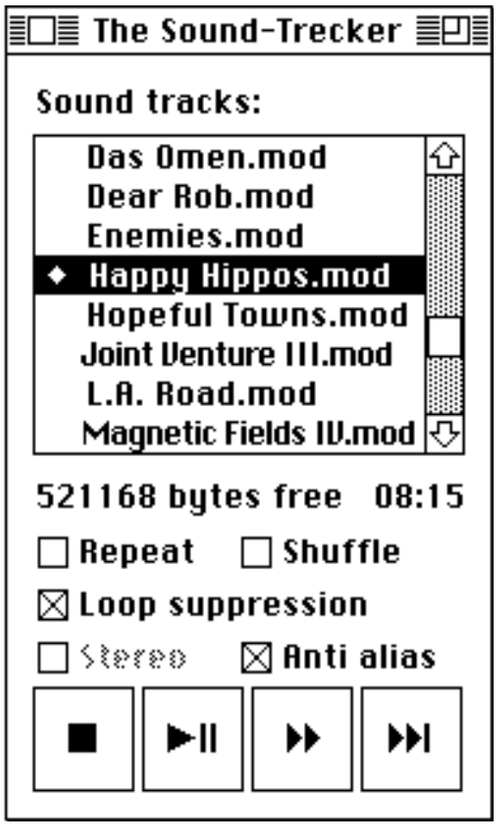


Figure 3.1: Reduced view of the Sound-Trecker

The Sound-Trecker: Documentation

Figure 3.1 shows the reduced view of the Sound-Trecker. By clicking the “zoom box” in the upper right corner the extended view may be activated. The Sound-Trecker usually starts up with the extended view, in this chapter, however, only the functions of the reduced view are discussed.

On top of the memory and playing time display the *track list* is located. In the track list you can compile your individual program of songs (as shown in chapter 5). The song being played is marked by a preceding 1 symbol. The option check boxes in the middle are explained in chapter 4. At the bottom four big control buttons are located.

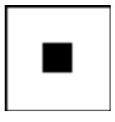
3.3 Playing a Song

To play a song, you must first place it into the track list. To do this, execute the menu command "Open…" in the "File" menu (shot cut ⌘-O, hold down the ⌘ key and depress O). After selecting the menu command, a standard file dialog will be presented, in which you should, as usual, choose the song to play. In addition to real sound track files the file dialog shows normal text files, since the sound track file type is usually at first set to text file after downloading from a mail box or copying from a PC disk (see chapter 2). When such a file is opened, the Sound-Tracker asks automatically, whether the file type should be adapted. Normally you will agree with that. If you open a real text file by mistake, the Sound-Trecker will notice this and reject the file as "no sound track".

When you open a sound track, it will be appended to the track list. If no song is being played, the new sound track will be loaded and started automatically. To start a song which is already in the track list, just double click on it.

3.4 The Control Buttons

The control buttons at the bottom of the window are used by simply clicking them with the mouse. They have the following purposes:



Stop: Clicking this button fades out and stops the song currently being played and removes it from memory (not from the track list);

Play/Pause: The currently played song is paused; clicking this button again restarts playing. If no song is being played, the song selected in the track list will be loaded and started. If the track list is empty, the "Open…" command is executed automatically.

Fast Forward: Play at double speed, as long as the button is held down (four times the speed if the option key is held down);

Next Song: the song currently being played is faded out. Then the next one is loaded and started.

Three of the buttons can be activated by menu commands in the “Miscellaneous” menu, so that the following keyboard equivalents are available. Press ⌘-. for the Stop-Button, ⌘-P for Play/Pause, and ⌘-+ for the next song.

4 Options

The Sound-Trecker offers a couple of options, which are the subject of this chapter.

4.1 Extended View

The Sound-Trecker normally starts up with extended view. It can be switched off and on by clicking on the “zoom box”.

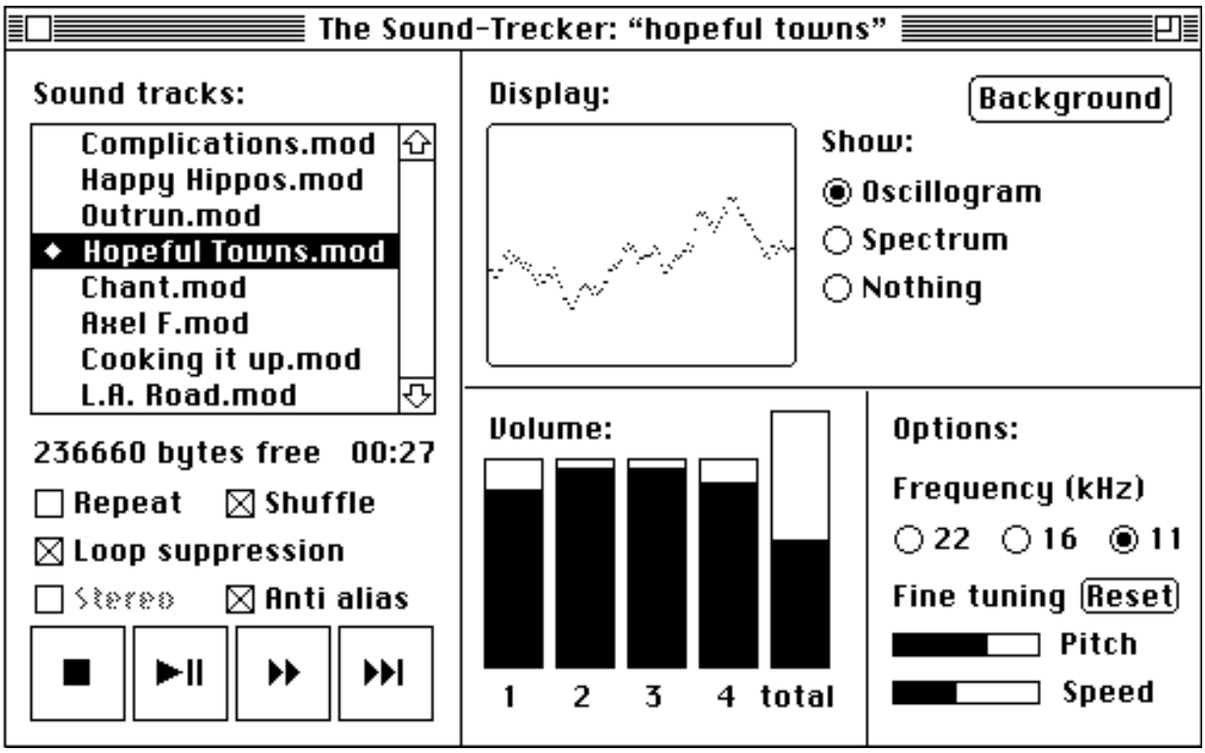


Figure 4.1: Extended view of the Sound-Trecker

4.2 The Options in Detail

To change options, several check boxes, buttons, and sliders are available. Most of the options can also be changed by the “Miscellaneous” menu respectively by keyboard short cuts.

Controlling the Playing Order

The playing order can be controlled by the following options:

Repeat (Short cut ⌘-R)**:** The program contained in the track list is usually played once. With the repeat option activated, the track list is started over again when the end is reached.

Shuffe: With the shuffle option activated, the songs are played in a random order.

Loop Suppression: Some sound tracks don't have a defined end. Instead, they run into an endless loop, which repeats the same bars again and again. If loop suppression is activated, a song falling into an endless loop will be faded out and stopped immediately. Then the Sound-Trecker continues with the next song.

Controlling the Playing Quality

The following options control the playing quality and indirectly the CPU time required to calculate the output data. When changing a quality option, the sound output may pause for a short time.

Stereo (⌘-M)**:** If your Macintosh is capable of producing stereo sound, you can use this check box to choose between stereo and mono sound.

Anti alias (⌘-A)**:** This option allows you to choose between two different sampling rate conversion algorithms (see section 6.1). If the check box is not checked, a simple and fast algorithm is used. This method produces a so-called “aliasing” distortion which may be significant in some songs, especially if you use a hifi amplifier and headphones. If anti-aliasing is active, a noticably better sound quality is achieved by linear interpolation. This algorithm, however, may increase the required CPU time by a factor of three.

Frequency (⌘-2: 22 kHz, ⌘-6: 16 kHz, ⌘-1: 11 kHz)**:** With this option you can select the sampling frequency (sampling rate) used for sound output. The maximum frequency component of the signal is half the sampling frequency, e.g., at a sampling rate of 11 kHz, the maximum frequency is 5.5 kHz (a little more than telephone quality). The required CPU time is proportional to the sampling rate; with 22 kHz the number of calculations is twice as high as with 11 kHz. With the most songs, the frequency does not make much difference in quality, especially, if you use the built in speaker. So usually 11 kHz should be enough.

Justage Options

You can use the two justage sliders to adjust the playing **pitch** and the **tempo** *independently* by about ±50%. The reset button resets the values to default.

The Volume Sliders

The music is being produced by mixing four sound channels. With the volume sliders the total volume as well as any channel volume may be changed. This enables you listen to each channel separately.

The total volume should be set to around 50% (as shown in Figure 4.1). The value should not be much greater, because then numerical ranges may be exceeded in calculations, which leads to clipping and a significant loss of quality. On the other hand, with a low volume the quantisation noise becomes more dominant: if you turn the total volume low, and turn up you amplifier, you will notice an increased noise.

The Background Button

The **background button** (⌘-H) is only available in System 7 (or later). It closes the Sound-Trecker window and activates one of the other running applications (e.g. the Finder). The music will be generated in the background. You can re-open the Sound-Trecker window via the application menu.

Under System 6.0.7 the Sound-Trecker window can be closed by clicking a window of another application while holding down the option key.

Graphical Display

The generated sound data may be displayed graphically, either as an oscillogram or a frequency spectrum. Figure 4.1 shows an oscillogram. The frequency spectrum shows the logarithmical absolute value over the linear frequency axis. The oscillogram as well as the spectrum, which requires a Fourier transform (FFT), are generated and displayed in real time (however, at 22 kHz with active anti-aliasing the display is quite slow on a Mac LC).

5 Compiling a Program of Songs

With the track list play programs of songs may be compiled. Such programs may be saved, along with all of the options, in a so called *configuration file* (Note, that the term “program” must not be confused with an executable program for the Macintosh).

5.1 Modifying the Track List

Adding a Song to the Track List

As described in section 3.3, a sound track is added to the track list by executing the menu command “Open…” in the “File” menu (short cut ⌘-O). Alternatively you can drag and drop sound track icons upon the Sound-Trecker icon in the Finder. With System 7, this works even, if the Sound-Trecker is already running. You may use this method, if you want to add many files to the track list, since the standard file dialog is quite slow. However, you should avoid dragging files with a wrong file type onto the Sound-Trecker icon, especially sound tracks whose file type has not yet been adapted (see section 2.3).

Selecting a Song

A song is selected by clicking on it with the mouse. The name of the selected file will be highlighted, as shown in figure 5.1 (the selected song is “Happy Hippos”).

Removing a Song From the Track List

To remove a song from the track list, first select it. Then press the delete key (the key may be labeled with “←”). Of course, the song is *not* removed from the Mac's disk, only from the track list.

Changing the Position of a Song

The position of a song in the track list simply by dragging the song to the new position and dropping it there. While the name is being dragged, a dotted frame as shown in figure 5.1 reveals the new position.

Figure 5.1: Dragging a song around in the track list

Duplicating a Song

If you want a song to appear more than once in the track list, you can duplicate the list entry with the menu command “Duplicate” in the “Edit” menu (key equivalent ⌘-D). A copy of the name appears just below the original name. Now you may drag the song to a new position.

5.2 Loading and Saving a Configuration

The current configuration (contents of the track list and all option settings) may be saved into a configuration file with the menu command “Save configuration” (⌘-S) or “Save configuration as…” in the “File” menu. The commands will present the usual standard file dialog to select the file name. Respectively, a configuration file can be loaded with the menu command “Load configuration” (⌘-L). You can also use “Open…” (⌘-O). In this case, the first song in the track list will be loaded and started automatically.

If you double-click on a configuration file in the Finder, the Sound-Trecker will be started. It will then load the configuration and start the first song. Under System 7 you may locate a configuration file (or an alias) in the “`Startup items`” folder (in the system folder). This will start up the Sound-Trecker in the background every time you boot or restart your Macintosh.

6 Technical Details

6.1 How the Sound-Trecker Works

Sound track files do not simply contain digitized music. This would waste to much storage space. For example, using a sampling rate of 22 kHz, digitizing would require 22 KByte/s, which is about 1.3 MByte/min! Current compression methods are unable of reducing the immense amount of data by more than 50% without a loss in quality.

When the Sound-Trecker plays a sound track, one part of the software simulates a (four voice) synthesizer, which is “played” by another part of the software. That's why sound track (“.MOD”) files contain two kinds of data:

- Data for synthesizer simulation. These data consists mainly of the digitized tone of real instruments. Only one note per instrument is sampled. It can then be played in different pitches.
- The “notes” which are used to “play” the synthesizer. Compared to the instrument data, the not data use up only a small part of the file, so that even long songs need less than 100 KByte, if they don't use too many instruments.

Mixing of the Four Channels

When playing a sound track, the Sound-Trecker's main task is to mix four sampled sounds together, each using a different pitch and volume, and transferring the result to the speaker. The Sound Manager of System 7 (also contained in System 6.0.7) is capable of doing exactly the same, so the implementation seemed to be straight forward at first. Unfortunately, Apple's Sound Manager is so slow, that my Mac LC has only enough processing time for three channels (but sound tracks require four).

That's why the Sound Trecker uses only *one* sound channel (`SndChannel`) of the Sound Manager, which is runs at a constant sampling rate of 22 kHz (the user can choose slower rates of 16 or 11 kHz to save processing time). The Sound-Trecker mixes the four voices by itself and creates one common data stream, which is then passed to the Sound Manager. By using optimized MC68020 assembler code the processing time could be reduced to approximately a third, so that even a Mac LC is capable of playing the music. However, due to specializing in the MC68020, there is no way of using the Sound-Trecker on MC68000 Macs.

Sampling Rate Conversion

In order to play an instrument in different pitches, the sampled sound is usually replayed with a sampling rate that differs from that used in recording the sound. This requires a sampling rate conversion, which takes the most of the processing time.

Lets look at an example. Assume, we digitized a “C” from a piano, and we want to play a “D” on a piano in a song. This means, we have to replay the sampled “C” at a slightly higher sampling rate. It's just like a tape recorder running a little to fast. The sampling rate has to be increased by $2^{1/12}$ per semitone, which is about 5.946%. “D” is two semitones higher than “C”, so the output sampling rate has to be approximately 11 kHz • 1.05946 • 1.05946 ≈ 12.347 kHz.

However, the Sound-Trecker generates its output using a fixed rate of 22 kHz, instead of the required 12.347 kHz. This means, that sample values are needed at times for which we do not have samples for. To estimate the required sample values, the Sound-Trecker uses one of two strategies, which can be selected by the anti alias button.

With anti aliasing off, the unknown sample value is simply replaced by the closest one known. This method is very fast, but it creates distortions which may be apparent with some instruments. These distortions are produced because of a violation of Shannon's sampling theorem.

Better output quality is achieved with the anti alias option switched on. In this case the unknown sample value is calculated from the next and the preceding known values by linear interpolation. The aliasing distortions are reduced significantly, although they don't disappear completely. To achieve this, a much more expensive algorithm would be necessary, which cannot be implemented on a MC68020 in real time.

6.2 Overview of the Sound Track (“.MOD”) File Structure

A sound track file consists of several sections. The first section contains information about the instruments. There are two variants of the format, which offer room for 15 respectively 31 instrument descriptions.

Each instrument description is 30 Bytes long and has the following structure:

```
struct FileInstrData {
    char Filename[22];          Original Amiga filename (irrelevant)
    unsigned int numWords;      Length of the sampled sound (in words!)
```

```
int volume;              Volume (0..64)
unsigned int loopWord;   Loop start (in words)
```

```
    unsigned int loopWords;     Loop length (in words)
};
```

The sampled data themselves are located at a different place in the file. The instrument description does *not* contain any information about *where* in the file the samples are located. This information has to be calculated from the instrument lengthes and other length information.

The next two sections of the file describe the note data. Notes are grouped in so called “patterns”. A pattern is a sequence of four bars of notes and can be used several times in a song. A table containing pattern numbers defines the order the patterns are to be uses. A single note description is 32 bits long and has the following data structure:

```
struct Command {
    int InstrHiNibble : 4;      Instrument number (Hi Nibble)
    int AmigaPeriod : 12; Output sampling rate = 3.579545 MHz / AmigaPeriod
    int InstrLoNibble : 4;      Instrument number (Lo Nibble)
    int EffectCmd : 4;          Special effect: Command code
    int EffectArg : 8;                          and one operand
};
```

A pattern is built of four channels with 64 notes each:

```
struct Pattern { struct Command Commands[64][4]; };
```

The pattern reference table has space for exactly 128 references (requiring 128 bytes). An additional value stored two bytes before the table indicates, how many entries of the table are actually to be played. The sampled data of the first instrument is stored immediately after the last pattern. Add the length fo the first instrument to this offset in order to obtain the location of the second instrument's samples. Add the second length to get the third sample offset etc. That's why the number of patterns stored in the file is of great importance. To calculate this number, determine the maximum of all pattern numbers in the pattern reference table and add 1. Make sure to calculate the maximum of *all* 128 table entries, not only the entries that are to be played! Now you might ask, who the hell has developped such a clumsy, perverted file format. The offender was a musician not very experienced in programming…

The Sound-Trecker: Documentation

The whole file structure is as follows:

- Original name of the song as a C string (20 bytes)
- Instrument descriptions (15*30 or 31*30 bytes)
- Number of pattern reference table entries to be played (1 byte)

- A completely useless value that should be ignored totally (1 byte)
- Pattern reference table containing up to 128 numbers of patterns to be played (128 bytes)
- Only in the 31 instrument variant: 31 instrument ID string ('`M.K.`' or '`FLT4`') (4 bytes)
- The patterns; the amount has to be calculated from the reference table (1024 bytes each)
- The sampled instrument data (number of bytes according to instrument description)

6.3 Playing Sound Tracks in Your Own Programs

It is possible to use Sound-Trecker routines from your own software. To enable this, the related program code was separated from the main program and stored in two code resources:

- ResType='`STrI`', ID=128: this resource contains the routines to load and interpret the sound track ("`.MOD`") files;
- ResType='`PSyn`', ID=128: the actual synthesizer routines are contained here. They generate the actual output data and pass them to the speaker via the Sound Manager.

Calling Mechanism

The Sound-Trecker routines are called via glue routines, which load the resources or find out their address in memory, and then jump to the corresponding code. The glue routines are available as a Think-C 4.0 compatible source code; the file names are "`'STrI' Glue.c`" and "`'PSyn' Glue.c`". The related data structures and function prototypes are contained in "`PSyn.h`" and "`STrI.h`". The example source code "`Simple Trecker.c`" contains a minimal sound track player which demonstrates the usage of the routines.

Short Description of the Routines

Below, all routines are discussed in short. Unfortunately, there is not enough space for a detailed description. You should use the example program as a base of your own development.

First, a sound channel must be allocated, just as when using Apple's Sound-Manager. The sound channel is described by a pointer to a `struct PChannel`:

```
struct PChannel * pc;
```

Allocate the channel:

```
err = OpenPChannel (4, stereoFlag, 445*2, &pc);
```

Now the sound track file has to be loaded into memory. First you must declare a handle to a `struct SoundTrack`:

```
struct SoundTrack ** strk;
```

Now load the file using the following call:

```
err = GetSoundTrack (vRefNum, fName, 0, &strk, FALSE);
```

The sound track obtained by this call must be linked to the sound channel:

```
LinkSoundTrack (strk, pc);
```

Start playing:

```
PChannelVolume (pc, -1, 0x8000);   Full volume (important!)
ResetPChannel (pc);                Reset internal counters to beginning of the song
err = StartPChannel (pc);          Start sound generation
```

While the music is being played, several options may be modified, see data structure '`struct musicRecord`' in '`STrI.h`'. Whenever anything is changed in the `musicRecord`, some internal variables have to be updated. Therefore call:

```
UpdateSoundTrack (strk);
```

Stop (or pause) the sound generation using:

```
StopPChannel (pc, fadeOutFlag);
```

After stopping, the music may be restarted by another call to `StartPChannel()`. When the `fadeOutFlag` is set, the music will be faded out, which will leave a zero main volume.

To remove the sound track from memory, stop the music with `StopPChannel()`, unlink the sound track from the sound channel, and the release all the memory occupied by the sound track:

```
UnlinkSoundTrack (strk);
DisposeSoundTrack (strk);
```

The Sound-Trecker: Documentation

I hope, that these short descriptions and the example source code will enable you to write simple applications of the Sound-Trecker routines. For further information, simply write me.

The Sound-Trecker: Documentation

Conditions of Using the Routines

The '`PSyn`' and the '`STrI`' resource as well as the source code "`Simple Trecker.c`" may be used only in public domain software under the following conditions:

- using the routines in public domain software is free of charge;
- commercial use is strictly forbidden without my explicit written agreement (my address is shown in the beginning of this documentation);
- the music routines in the resource are *not* free of copyrights. That's why any public domain program using the routines, as well as its documentation, must contain a copyright note for the two resources including my address;
- a copy of the program has to be sent to me;
- the resources may not be modified (if you discover any bugs, please tell me to remove them. I want to avoid too many different versions).