NAME

TiMidity − MIDI to WAVE converter and player

SYNOPSIS

timidity [−options] filename [filenames ...]

DESCRIPTION

TiMidity is a MIDI to WAVE converter using Gravis Ultrasound−compatible patch files to generate digital audio data from General MIDI files. The data can be stored in a file for processing, or played in real time through an audio device.

OPTIONS

The following command line options are accepted by version 0.2i of TiMidity:

−o *filename*

Place output on *filename*, which may be a file, device, or HP-UX audio server, depending on the output mode selected with the **−O** option. The special filename ‘‘−’’ causes output to be placed on stdout.

−O *mode*

Selects the output mode from the compiled-in alternatives. *mode* must begin with one of the supported output mode identifiers. Run TiMidity with the **−h** option to see a list. The following identifiers should be available in all versions:

−Or Generate raw waveform data. All format options are supported. Common formats include:

−OrU uLaw

−Or1sl 16-bit signed linear PCM

−Or8ul

8-bit unsigned linear PCM

−Ow Generate RIFF WAVE format output. If output is directed to a non-seekable file, or if TiMidity is interrupted before closing the file, the file header will contain 0xFFFFFFFF in the RIFF and data block length fields. The popular sound conversion utility *sox* is able to read such malformed files, so you can pipe data directly to sox for on−the−fly conversion to other formats.

Format options

Option characters may be added immediately after the mode identifier to change the output format. The following options are recognized:

8 8-bit sample width

1 16-bit sample width

l Linear encoding

U uLaw (8-bit) encoding

M Monophonic

S Stereo

s Signed output

u Unsigned output

x Byte-swapped output

Note that some options have no effect on some modes. For example, you cannot generate a byte-swapped RIFF WAVE file, or force uLaw output on a Linux PCM device.

–s *frequency*
Sets the resampling frequency. Not all sound devices are capable of all frequencies -- an approximate frequency may be selected, depending on the implementation.

–a Turns on antialiasing. Samples are run through a lowpass filter before playing, which reduces aliasing noise at low resampling frequencies.

–f Toggles fast envelopes. Some MIDI files sound better when notes decay slower -- it gives the impression of reverb, which TiMidity doesn't currently support.

–e Make TiMidity evil. On Win32 version, this increases the task priority by one. It can give better playback when you switch tasks at the expense of slowing all other tasks down.

–p *voices*
Sets polyphony (maximum number of simultaneous voices) to *voices*.

–A *amplification*
Multiplies the master volume by *amplification%*.

–C *ratio*
Sets the ratio of sampling and control frequencies. This determines how often envelopes are recalculated -- small ratios yield better quality but use more CPU time.

–L *directory*
Adds *directory* to the library path. Patch, configuration, and MIDI files are searched along this path. Directories added last will be searched first. Note that the current directory is always searched first before the library path.

–c *file* Reads an extra configuration file.

–I *number*
Uses the program *number* as the default instrument. Any Program Change events in MIDI files will override this option.

–P *file* Uses the patch *file* for every program except drums. Program Change events will be ignored. This is useful for testing new patches.

–D *channel*
Marks *channel* as a drum channel. If *channel* is negative, channel –*channel* is marked as an instrumental channel. If *channel* is 0, all channels are marked as instrumental.

–Q *channel*
Causes *channel* to be quiet. If *channel* is negative, channel –*channel* is turned back on. If *channel* is 0, all channels are turned on.

–F Turns on fast panning to accommodate MIDI pieces that expect panning adjustments to affect notes that are already playing. Some files that don't expect this have a habit of flipping balance rapidly between left and right, which can cause severe popping when the -F flag is used.

–U Instructs TiMidity to unload all instruments from memory between MIDI files. This can reduce memory requirements when playing many files in succession.

–i *interface*
Selects the user interface from the compiled-in alternatives. *interface* must begin with one of the supported interface identifiers. Run TiMidity with the **–h** option to see a list. The following identifiers may be available:

–id The dumb interface -- plays files in sequence, prints messages according to verbosity level. The trace mode shows the current and total playing time.

–im The motif interface -- X Window interactive interface.

–ik The Tcl/Tk interface -- X Window interactive interface.

–in The ncurses full–screen interface with interactive controls.

−is The S-Lang full−screen interface with interactive controls.

Interface options

Option characters may be added immediately after the interface identifier. The following options are recognized:

v Increases verbosity. This option is cumulative.

q Decreases verbosity. This option is cumulative.

t Toggles trace mode. In trace mode, TiMidity attempts to display its current state in real time. For the Linux sound driver, this is accomplished through the use of short DMA buffer fragments, which can be tuned via the **−B** option.

−B *fragments*

For the Linux sound driver, selects the number of buffer fragments in interactive mode. Increasing the number of fragments may reduce choppiness when many processes are running. It will make TiMidity seem to respond sluggishly to fast forward, rewind, and volume controls, and it will throw the status display off sync. Specify a *fragments* of 0 to use the maximum number of fragments available.

FILES

TiMidity looks for the configuration file *timidity.cfg* at startup, before processing any options. If it can't be accessed, and the library path is changed with a **−L** option on the command line, then the default file will be sought again along the new library path after processing all options, unless another configuration file was specified with the **−c** option.

Configuration files define the mapping of MIDI programs to instrument files. Multiple files may be specified, and statements in later ones will override earlier ones. The following statements can be used in a configuration file:

dir *directory*

Adds *directory* to the search path in the same manner as the **−L** command line option.

source *file*

Reads another configuration file, then continues processing the current one.

bank *number*

Selects the tone bank to modify. Patch mappings that follow will affect this tone bank.

drumset *number*

Selects the drum set to modify. Patch mappings that follow will affect this drum set.

number file [options]

Specifies that the the MIDI program *number* in the current tone bank or drum set should be played using the patch *file*. *options* may be any of the following:

amp=*amplification*

Amplifies the instrument's volume by *amplification* percent. If no value is specified, one will be automatically determined whenever the instrument is loaded.

note=*note*

Specifies a fixed MIDI note to use when playing the instrument. If *note* is 0, the instrument will be played at whatever note the Note On event triggering it has. For percussion instruments, if no value is specified in the configuration file, the default in the patch file will be used.

pan=*panning*

Sets the instrument's default panning. *panning* may be **left**, **right**, **center**, or an integer between -100 and 100, designating full left and full right respectively. If no value is specified, the default in the patch file will be used. Note that panning controls in MIDI files will override this value.

keep={**loop**|**env**}
By default, percussion instruments have their loop and envelope information stripped. Strangely shaped envelopes are removed automatically from melodic instruments as well. **keep** can be used to prevent stripping envelope or loop data. For example, the Short and Long Whistle percussion instruments (General Midi numbers 71 and 72) need to have ‘‘**keep=loop keep=env**’’ specified in the configuration file.

strip={**loop**|**env**|**tail**}
Force removal of loop or envelope information from all patches in the instrument, or strip the tail, i.e. all data after the loop. Some third-party instruments have garbage after the loop, as evidenced by a clicking noise whenever the instrument is played, so adding the **strip=tail** option will markedly improve sound quality.

NOTE: Whenever any filename ends in one of the compiled−in compression identifiers, such as *.gz*, or *.sht*, TiMidity will pipe the file through the appropriate decompressor. MIDI files often compress very well, so the ability to handle compressed files can be useful.

The special filename ‘‘−’’ can be used on the command line to indicate that a MIDI file should be read from stdin.

COPYRIGHT

AVAILABILITY

The latest release is available on the TiMidity Home Page, URL *http://www.clinet.fi/˜toivonen/timidity/*.

BUGS

8-bit and low-rate output sounds worse than it should.

Eats more CPU time than a small CPU-time-eating animal.

AUTHORS

Tuukka Toivonen <toivonen@clinet.fi>
HP−UX audio code, X-Motif interface, icons and antialiasing filter by Vincent Pagel <pagel@loria.fr>
Tcl/Tk interface by Takashi Iwai <iwai@dragon.mm.t.u-tokyo.ac.jp>
Windows 95/NT audio code by Davide Moretti <dmoretti@iper.net>
DEC audio code by Chi Ming HUNG <cmhung@insti.physics.sunysb.edu>
S−Lang user interface by Riccardo Facchetti <riccardo@cdc8g5.cdc.polimi.it>