Default

Håvard Pedersen

COLLABORATORS

	TITLE : Default	

ACTION	*NAME*	*DATE*	*SIGNATURE*
WRITTEN BY	Håvard Pedersen	January 17, 2023	

REVISION HISTORY

NUMBER	DATE	DESCRIPTION	NAME

Contents

1 Default **1**

1.1 ProTracker documentation . . . 1
1.2 Introduction . . . 2
1.3 What is ProTracker? . . . 2
1.4 Legal mush . . . 2
1.5 How to start ProTracker . . . 3
1.6 Gadgets . . . 3
1.7 Main screen - Various . . . 4
1.8 Main screen - Song section . . . 6
1.9 Main screen - Sample section . . . 8
1.10 Main screen - Edit options . . . 10
1.11 Main screen - Sample edit . . . 11
1.12 Main screen - Chord editor . . . 14
1.13 Main screen - Filter editor . . . 14
1.14 Main screen - Edit gadgets . . . 15
1.15 Setup screen - Miscellaneous . . . 16
1.16 Setup screen - Flags . . . 20
1.17 Disk operations . . . 22
1.18 Sampler - Main . . . 25
1.19 Sampler - Volume requester . . . 29
1.20 Nice to know . . . 30
1.21 Pointer colors . . . 30
1.22 Keyboard . . . 30
1.23 Effect commands . . . 34
1.24 Arpeggio effect command . . . 35
1.25 Slide up effect command . . . 35
1.26 Slide down effect command . . . 36
1.27 Toneportamento effect command . . . 36
1.28 Vibrato effect command . . . 36
1.29 Toneportamento and volumeslide effect command . . . 37

1.30 Vibrato and volumeslide effect command . . . 37
1.31 Tremolo effect command . . . 37
1.32 Sample offset effect command . . . 38
1.33 Volume slide effect command . . . 38
1.34 Position jump effect command . . . 38
1.35 Set volume effect command . . . 38
1.36 Pattern break effect command . . . 39
1.37 Adjust tempo effect command . . . 39
1.38 Set filter miscellaneous effect command . . . 39
1.39 Fineslide up miscellaneous effect command . . . 40
1.40 Fineslide down miscellaneous effect command . . . 40
1.41 Glissando control miscellaneous effect command . . . 40
1.42 Vibrato control miscellaneous effect command . . . 41
1.43 Set finetune miscellaneous effect command . . . 41
1.44 Pattern loop miscellaneous effect command . . . 41
1.45 Tremolo control miscellaneous effect command . . . 42
1.46 Retrig note miscellaneous effect command . . . 42
1.47 Fine volumslide up miscellaneous effect command . . . 42
1.48 Fine volumslide down miscellaneous effect command . . . 42
1.49 Cut note miscellaneous effect command . . . 43
1.50 Delay note miscellaneous effect command . . . 43
1.51 Delay pattern miscellaneous effect command . . . 43
1.52 Invert loop miscellaneous effect command . . . 44
1.53 Requesters and I/O . . . 44
1.54 Standard paths . . . 44
1.55 Keyboard percussion . . . 45
1.56 What is DYN samples? . . . 45
1.57 How to create music using PT . . . 46
1.58 What is a module? . . . 46
1.59 Patterns . . . 47
1.60 Position table . . . 48
1.61 Samples . . . 48
1.62 Miscellaneous . . . 50
1.63 Chronicles and anecdotes . . . 50
1.64 Development and progress . . . 50
1.65 Other tracker-clones . . . 53
1.66 Version history . . . 55
1.67 V0.89 BETA . . . 56
1.68 V1.0A . . . 57

1.69 V1.0B . . . 57
1.70 V1.0C . . . 57
1.71 V1.1A . . . 58
1.72 V1.1B . . . 60
1.73 V1.2 . . . 61
1.74 V1.2C . . . 61
1.75 V1.3B . . . 61
1.76 V1.8 . . . 62
1.77 V2.0 . . . 62
1.78 V2.1A . . . 62
1.79 V2.2A . . . 64
1.80 V2.3A . . . 65
1.81 V3.00 BETA . . . 65
1.82 V3.01 . . . 67
1.83 V3.10 BETA . . . 68
1.84 V3.14B . . . 68
1.85 V3.15 . . . 68
1.86 Known bugs . . . 69
1.87 Things we'll never implement . . . 71
1.88 Things we will implement . . . 71
1.89 Frequently asked questions . . . 72
1.90 Thanks and acknowledgements . . . 74
1.91 Contacting the authors . . . 74
1.92 For the technical minded . . . 74
1.93 ProTracker song&module format . . . 75
1.94 ProTracker IFF song&modules . . . 78
1.95 Using the PT playroutine . . . 83
1.96 Using the PPR playroutine . . . 85
1.97 Using SoundFX . . . 89
1.98 Using PTCalcTime . . . 90
1.99 The TrackerTool command . . . 91
1.100ProPacker v2.1 . . . 92
1.101PPRStrip & PPRSampler . . . 93
1.102CIA tempo calculation . . . 94
1.103DYN Samples . . . 94
1.104Index . . . 97

Chapter 1

Default

1.1 ProTracker documentation

```
                    ProTracker V3.x - The AmigaGuide documentation


    "...The use of noise to make music will continue and increase until
    we reach a music produced through the aid of electrical
    instruments, which will make available for musical purposes any and
    all sounds that can be heard."
                    -- John Cage

    "It is not hard to compose, but it is wonderfully hard to let the
    superfluous notes fall under the table."
                    -- Johannes Brahms


    Table of contents:


                Introduction

                Gadgets

                Nice~to~know

                How~to~create~music~using~ProTracker

                Miscellaneous

                For~the~technical~minded
                Powerpacker.library is © Nico François and req.library is © Bruce ↩
                   Dawson
and Colin Fox.

This AmigaGuide® help-file is based on the printable help-file for
ProTracker V1.3 by Lars Hamre, and is made by Håvard "HOWARD/MENTAL
DISEASES" Pedersen.
                DYN~technical~chapter
                 by PUW.
```

```
More specific credits are found in both the '
                version~history
                ' and the
'
                contact~the~authors
                ' chapters.

Consider this entire documentation a beta. If there's anything missing,
please
                inform~us
                .
```

1.2 Introduction

```
                This chapter contains basic information about ProTracker and how  ↩
                   and why
to get started using it.


                What~is~ProTracker?

                Legal~mush

                How~to~start~ProTracker
```

1.3 What is ProTracker?

ProTracker is a so-called "Tracker", a program designed for creating music savable as a MOD-file. The most significant difference between ProTracker and other trackers are that ProTracker is based on the original tracker, SoundTracker. It has also more built-in tools than almost any other tracker and supports all aspects of the MOD file-format. (With the exception of 100-pattern modules)

If you doesn't count MIDI, about 98% of all music on the Amiga is made on ProTracker. The ProTracker has become a standard for music and the MOD-files are even supported by most sample-based music programs on the PC. (I've even seen some players for Macintosh that supported them!)

We must admit that the user-interface for the ProTracker is neither multitasking-friendly nor does it conform to the standard GUI guidelines. The reason for this is that the GUI consumes quite a bit of the program itself, and it would be a very time-consuming job to port it to use the system. (Besides, not all keyboard-shortcuts are comfortably read through the system.)

1.4 Legal mush

All users and distributors must read this!

- This program is fully public domain software, and may therefore freely~be~copied by anyone to anyone.

- The authors will not accept any charges demanded for distribution~and/or~copying of this program which exceeds the nominal charge for postage,~handling~and disks.

- The authors takes no responsibility for any damages caused by use or~misuse~of this program, though all efforts are taken to make it as faultless~as~possible.

- This program is law-established intellectual achievement, and is~therefore~regulated by laws concerning such, where such laws exists.

- By using and/or distributing this program, you indicate your agreement~with~these paragraphs.

- If there are any of these paragraphs you do not understand, you are~obliged~to contact the
authors
and ask for further information
before~indicating your~agreement through the previous paragraph.

- Any violation of these paragraphs is both illegal and immoral.

1.5 How to start ProTracker

ProTracker may be started from both CLI and Workbench. (Type name in CLI or doubleclick on icon from Workbench)

No arguments or tooltypes are supported yet, though.

1.6 Gadgets

This chapter should explain all gadgets and gizmos in ProTracker.

Mainscreen

Various

Song~section

Sample~section

Edit~options

Sample~edit

Chord~editor

Filter~editor

Edit~gadgets
Setup

Miscellaneous

Flags
Disk operations

Disk~operations
Sampler

Main

Volume~requester

1.7 Main screen - Various

CLOSEGADGET

This is a small box located in the upper left corner of the ProTracker screen. Pressing this will quit.

QUADRASCOPE

This is 4 "boxes" located in the upper right part of your screen. These shows the waveform currently being played on each channel. Clicking in any of these will disable the respective channel and ghost the box.

DEPTHGADGET

This is another box located in the upper right corner of the ProTracker screen. Pressing this will push the ProTracker screen behind all other screens currently open.

DATE

Shows the current date. (Atleast your computers idea of it!)

TIME

Shows the system time.

PLAY

Shows the time elapsed since ProTracker started playing.

EDIT OPTIONS (Numbers 1 - 7)

The edit options represent several small "tools" grouped into (somewhat) clear categories. The upper line will state what these functions will affect. The numbers are:

1 -
Edit~options
2 -
Sample~edit
3 -
Chord~editor
4 -
Filter~editor
STATUS

Shows a somewhat descriptive text describing what ProTracker is doing.

CHIP

Shows how much chip-memory there is left on your Amiga.

FAST

Shows how much fast-memory there is left on your Amiga.

TUNE

Shows how much memory the current module uses.

PUBL

Shows how much memory (any type) there is left on your Amiga.

PATTERNNUMBER GADGET

This is a small box to the left of the BPM gadget, and represents the number of the pattern currently being edited. Click on it to type in a new number.

BPM

The tempo gadget on the status bar is for setting the CIA speed, if CIA timing is used. The gadget will be updated every time you set the speed using the
F~command
(if CIA that is). This gadget does not show actual beats per minute unless the speed is 6!

The main screen has some indicators on the left side of the statustext. The indicators are as follows:

M S M (Metronome ON, Split keyboard ON, Multi keyboard ON) I 0-9 (AutoInsert ON, AutoInsert Macro)

1.8 Main screen - Song section

SONGNAME

This defines the name of your module.

MASTERVOLUME

Defines the volume of your module. This gadget affects playing only, and the setting is not saved along with your module in current version of ProTracker.

POS

Defines the current position in the position-table.

I(NSERT)

Insert a position into your song.

D(ELETE)

Delete a position from your song.

PATTERN

Defines which pattern will be played at the selected position.

LENGTH (SONG)

Defines the length of the song.

PLAY

Will play the song from the current position in the song. The pointer turns yellow, just to show you what’s going on.

PATTERN

Will play the current pattern which is shown at the bottom of the screen. The pointer turns yellow here as well. Holding down the right button while pressing play,pattern or record will play from the current patternposition.

CLEAR

Will first ask you what you want to clear. You can clear either all, song or samples. In addition to the gadgets you can use "A" for All, "O" for Song, "S" for samples and "C" or ESC for Cancel.

STOP

Will stop playing of songs and patterns, recording, and will turn edit mode off.

RECORD

Will put you in edit mode, but also play the current pattern or song. You can select this in the Edit Options menu. While the pattern or song is playing, you can type in notes and numbers from the keyboard, and they will appear in the pattern as it scrolls. The notes and numbers will also be quantized to the nearest slot, so that keeping a steady rhythm is no problem. The pointer will turn blue here as well as in the the normal edit mode.

SETUP

Will go to the
Setup~Screen
.

CONTINUE

Will continue the module from the last stopped position.

EDIT

Will put you in edit mode. The pointer turns blue, and you can enter notes and numbers from the keyboard. Use the arrowkeys to move up/down and left/right in the pattern. Entering a note or a number will cause the pattern to jump one or more slots down.

DISK OP.

Will go to the
Disk~operation~screen
.

1.9 Main screen - Sample section

SAMPLENAME

This defines the name of the current sample.

SAMPLE

The number of the chosen sample. You can have up to 31, or hex $1F samples in a song. Pressing both mousebuttons at the same time will set the samplenumber to zero. You can then record the pattern with sample 0 to prevent ProTracker from setting the volume each time you play a new note.

VOLUME

Use this to set the volume the current sample will be played with.

FINETUNE

Tune your untuned samples to match the others.

```
    0 436.4 hz    -1 432.1 hz
    1 439.0 hz    -2 429.6 hz
    2 441.6 hz    -3 426.3 hz
    3 445.1 hz    -4 423.1 hz
    4 447.8 hz    -5 419.9 hz
    5 451.5 hz    -6 416.7 hz
    6 455.2 hz    -7 414.4 hz
    7 457.0 hz    -8 412.0 hz
```

To experienced musicians, this table clearly will seem wrong. A normal tuning should be 440! The reason to this is that ProTracker uses NTSC periods for playback.

LENGTH (SAMPLE)

The Length gadgets are simply used for setting the length of the sample. A sample can be up to 64k, or $fffe long. You can add workspace behind the sample by increasing the length and letting go of the button. ProTracker will ask if you are sure, and if you are, allocate more memory for the sample.

REPEAT

Here you set the start of the sampleloop.

REPLEN

Here you set the length of the sampleloop.

LOAD SAMPLE

Will simply try to load the current samplename. Use this when you've fucked up in the sample editor, and have destroyed a sample.

KILL SAMPLE

Will remove the current sample from your module.

SAMPLER

Will go to the
Sampler~screen

.

1.10 Main screen - Edit options

QUANTIZE

Will move the notes you record to every n'th slot. Entering 00 will turn off the quantizing, and the notes you play will always be inserted at the patternposition you are at. Entering 01 will quantize the notes the the nearest slot according to the speed. i.e. if you play a note after the first half has been played, it will be quantized to the slot below. Entering a value like 8 will quantize to every 8th note, and so on. Got that?

METROSAMP

[This option is not currently implemented, nor documented!]

METRONOME

A metronome is a steady sound which helps you keep the speed when recording. (I'm not particulary good at explaining, try it out!)

The first number is the speed of the metronome, and the second is the channel to play it on. The Sample used for metronomes is always sample $1F. Load your own favourite metronome sample. The metronome will always be played at C-3, but you can still change the volume and loop values. To turn off the metronome, just set the speed or channel to 0.

MULTI

This table is used with the multi keyboard option. The four numbers represent what channel each channel will jump to next. 1-2, 2-3, 3-4 and so on.

RECORD HOLD

When this is set to "on", record will not start until the first keypress.

SAMPLE

This selects wether the edit buttons should affect all samples, or the current samples only.

RECORD

This selects wether the record function should record a pattern, or the entire song.

PLAYNOTE

When set to "multi", ProTracker will jump to another channel after you play a note on the keyboard. This makes it possible to play two or more notes at the same time (very useful with midi).

EDIT

This selects wether ProTracker should go right when inserting notes, or down to the next step.

1.11 Main screen - Sample edit

To the left of the title bar is a box which states what should be affected by the copy, exchange and delete gadgets. (Current track, whole pattern, real samples.)

To the right of the title bar is a box which states the mixingmode. "half" will halve the volume when mixing and echoing to avoid clipping, while "clip" will not halve the volume and possibly clip the sample.

DELETE

Will delete all notes with the current sample in current track or whole pattern.

KILL

Will kill the current sample. That is, remove it from memory and reset all sample settings. It will not be deleted from the track or pattern.

EXCHANGE

Will exchange the samplenumber shown in the "from" gadget with the samplenumber in the "to" gadget and vice versa.

COPY

Will move the samplenumber shown in the "from" gadget to the sample-
number in the "to" gadget.

MIX

Will mix one sample with another. ProTracker asks you which two samples
to be mixed, and where to put the result.

Holding the right button and pressing mix will mix the current sample
with itself. You can offset the sample by setting a position in the
"pos" gadget If you set "mod" to a non-zero value, the sample will also
be modulated.

ECHO

Will create a echo effect on the current sample. Use "pos" to set the
delay time of the echo. If you want more room to echo in, just turn up
the length of the sample.

BOOST

Will turn up the treble of the sample. Use this on hi-hats and snares!
If the
sampler~screen
is open and a range is selected, this will only
affect the selected range.

FILTER

Will Delta-filter the sample. Use this on noisy basses. If the
sampler
screen is open and a range is selected, this will only affect the
selected range.

X-FADE

Will crossfade the sample (mix with itself, backwards). Handy for
looping samples that are hard to loop.

BACKWD (Backwards)

Will turn the sample backwards!

UPSAMPLE

Will remove every second byte of the sample, halving the length and
shifting the pitch one octave up.

DOWNSAMPLE

Will double every byte of the sample, doubling the length, and shifting
the pitch one octave down.

POS

This is just an offset in the sample, used for a lot of things.This one
has a numbergadget as well. Holding the right mousebutton while pressing
the numbergadget will zero the value.

MOD

This is used for modulation. Press "mod" to modulate the current sample.
Holding the right button while pressing the numbergadget will zero the
value.

CUTBEG(INNING)

Will chop the number of bytes set in the "pos" gadget off the beginning
of the sample.

FU (Fade Up)

Will fade the volume from 0 to 100%. Use "pos" to select where in the
sample to fade up to. If the
                sampler~screen
                 is open and a range is
selected, this will only affect the selected range.

FD (Fade Down)

Will fade the volume from 100 to 0%. Use "pos" to select where in the
sample to fade down from. If the
                sampler~screen
                 is open and a range is
selected, this will only affect the selected range.

VOL

With this you can change the "real" volume of the sample. Just set a percentage and press "vol". vol has a numbergadget. Holding the right button while pressing it will set the value to 100%. If the sampler screen is open and a range is selected, this will only affect the selected range.

You may also set "POS" by clicking on the sample and setting the cursor-line.

1.12 Main screen - Chord editor

FROM SAMPLE

Selects which sample to mix the chords from. Clicking on it sets the current sample.

TO SAMPLE

Selects which sample to put the chord into. Clicking on it sets the current sample.

NOTES

Sets how many notes your chord contains.

ADJUST

[Sorry, not enough info for documentation...]

MAKE CHORD

Does the work.

NOTE 1 - NOTE 7

The notes to use in the chord. Click in gadgets to edit.

1.13 Main screen - Filter editor

[No documentation exists on the filter editor, but I couldn't get it to do anything!]

1.14 Main screen - Edit gadgets

TRACK/PATT/CMDS/BLOCK

Selects what to be affected by the edit gadgets. (Refered to as range.)

INSERT

Inserts the buffer at current position.

CUT

Deletes the range and moves it to the buffer.

DELETE

Deletes the current step and moves all preceding one step up. (Not block)

COPY

Copies the range to the buffer.

CLEAR

Deletes the range. (Not block)

PASTE

Pastes the buffer out to current position.

FLIP

Turns the range backwards.

EXCHANGE

Exchange range with buffer.

OCTAVE UP

Transposes range one octave up.

OCTAVE DOWN

Transposes range one octave down.

NOTE UP

Transposes range one note up.

NOTE DOWN

Transposes range one note down.

SCROLL UP

Scrolls range one step up.

SCROLL DOWN

Scrolls range one step down.

ROTATE UP

Same as scroll, but inserts note scrolled out at top at bottom step.

ROTATE DOWN

Same as scroll, but inserts note scrolled out at bottom at top step.

1.15 Setup screen - Miscellaneous

MAIN MENU

Will exit the setup screen, and return to the
main~screen
.

LOAD CONFIG

Will load the selected config file.

SAVE CONFIG

Will save the selected config file.

CFG

Selects the current configuration to use.

RESET ALL

Will reset to the original ProTracker configuration.

THE COLOR PALETTE

The color palette is simple to use. Just select a color, and use the R, G and B sliders to set the color. You may also edit the colors for the VU-meters by pressing anywhere within the VU-cols and edit.

COPY

Copies a color. Select the color to copy, select "copy", select destination color.

SPREAD

For use with the VU-meter colors. Press at start color, select "spread", and select destination color to make ProTracker calculate a nice fade between the two colors.

ARROWS (UP AND DOWN)

Rotates the colors for the VU-meter.

SWAP

Swaps two colors. Use as copy.

CANCEL

Will set the last saved colors.

UNDO

[Does not work, sorry!]

DEF1 - DEF6

[Does not work, sorry!]

TEMPO

This is where you set your default CIA timing tempo. Range: 32-255.

SPEED

This is where you set your default Vblank timing speed. Range: 01-FF.

TIMING

We included this so that American users also could enjoy ProTracker, and wouldn't have to use sonix or any other terrible music program...

You can choose between CIA or Vblank timing. Vblank is the timing-method soundtrackers have been using since the dawn of time, while CIA is a much better and
accurate~timing
with the tempo measured in beats per
minute. Using Vblank on NTSC amigas will cause the song to play 20% faster. With CIA, there's no difference.

SOFTINT

[No documentation exists, but audible problems have occured when using "on".]

SCREEN

This enables you to move the screen to fit your overscan settings.

TUNENOTE

Selects which note to use for the tuning tone in the sample editor.

TUNE VOL

Selects the volume for the tuning tone.

RECOVER SONG

[Not implemented]

CLEAR

Will clear the splits. (See below)

SPLIT

You can set 4 splits on the keyboard, each with it's own sample, splitpoint and transpose. Just type in a sample number and select the key to split at by pressing the appropriate one. The transpose note for each split is the first note in that split-range. Notes below the first split will be played with the current sample. Split is great for recording drums, or for playing untuned samples in tune (use together with finetune).

ACCIDENTAL

Simple enough, accidental allows you to select sharp (#) or flat (b) notes. This will not be saved with the song!

PRINT SONG

Will print the song to the path shown below the "Print Song" gadget. The

print path can be 31 chars long. (To make it actually appear on your printer, use "PRT:")

PP EFF

Selects which crunchmode to use when crunching with powerpacker library. The name of the modes explain it all.

VUMETER

Selects which VU-meter mode to use. "Fake" is the normal VU-meter used in all Trackers from the beginning of time, but "real" gives a much better impression of the actual volume of the channel.

1.16 Setup screen - Flags

SPLIT

Toggles between normal and split keyboard.

TRANS(POSE)DEL(ETE) ON/OFF

When on, notes transposed out of range will be deleted.

SOLIDSCOPE

When on, the quadrascopes will be rendered in a somewhat fancier way.

BLANKZERO

When off, ProTracker will show the patterndata with zeroes where no changes take place. When set to on, these places will be rendered blank. (Well, sort of, anyway!)

SHOWDEC(IMAL)

When on, memory sizes will be shown in decimal.

FILTER

Toggles the low-pass filter of the Amiga. (Not available on all Amigas)

NTSC

When on, ProTracker will only be visible 200 lines at a time, and scroll in order to be able to display the entire screen. This is only useful for users using a NTSC based video system.

LACED

When on, ProTracker will use the interlace (flickering) mode of the Amiga in order to make all genlocks a little bit happier. 8)

UNDYN IFF

[No documentation available at time of writing.]

UNDYN MOD

[No documentation available at time of writing.]

OVERRIDE

When on, ProTracker will ignore any paths or disknames when loading a song. All the samples will be loaded from the current sample path.

NOSAMPLES

When on, ProTracker won't load the samples when loading a song or module.

SHOW DIRS

When on, directories will be shown in the
disk~operations~screen
.

LOAD LOOP

When on, ProTracker will load loops from IFF-samples.

AUTODIR ON/OFF

When on, ProTracker reads the current directory path automatically when
the
 is opened.

AUTOEXIT ON/OFF

When on, ProTracker will automatically exit from the
disk~operations
when loading a song, module, track or pattern.

MULTICACHE

[No documentation available at time of writing.]

CUTTOBUFF

If enabled, all cut operations will copy to the paste-buffer.

CUTTOLOOP

[Not implemented]

UPS>28kHz

[Not implemented]

1.17 Disk operations

The disk operations screen has a directory window at the right ↩
edge of
the screen, were you can select which files to load.

MAIN MENU

Will exit disk operations and return to the
main~screen
.

PACK

Selects wether modules and samples should be packed using powerpacker library.

SAMPFORM

Selects wether samples are to be saved as IFF 8SVX or as RAW 8-bit data.

SAVE ICON

[Not implemented]

HIDE .INFO

Selects wether ProTracker should hide .info-files or not. (Icon files used by the Workbench.)

RESET PATHS

Resets the paths to the internal paths for loading.

FORMAT DISK

Formats the disk specified by the path gadget. Should be as easy to use as the format command on your Workbench-disk. If you don't know which options to use, just click on "format" and wait...

RELABEL DISK

Changes the name of a disk. (DF0:?)

MASK

Specifies which files are to be shown in the filerequester.

PATH

This shows the current path. The paths for each file-type is saved in the config file on the setup screen.

FILE

This shows the current filename.

MAKEDIR

Creates a directory.

RENAME FILE

Renames a file.

DELETE FILE

Deletes a file.

Here follows a lot of "mode"-like gadgets. Clicking in the small box to the left of these gadgets (or the big gadget next to the file-name gadget) uses the path and mask selected for the filetype and enables loading of the selected format. Clicking direct in the mode-gadget selects saving.

SAVE EXE

[Not implemented]

SAVE MODULE

Saves the current module.

SAVE SONG

Saves the current module as a song. The sampledata for the instrument is not saved, so ProTracker will have to re-read these when reading the song back to memory. This way of storing modules is considered obsolete, and only loading of such files is recommended.

SAVE INSTR

[Not implemented]

SAVE SAMPLE

Saves the current sample.

SAVE PATT

[Not implemented]

READ DIR

Re-reads the selected path.

PARENT

Skips to the parent directory of your current path.

FREE

Shows amount of bytes free in the selected path.

All other gadgets on the disk operation screen is devices and assigns on
your system and may be pressed in order to make ProTracker make the
device/assign the current path. Pressing right button in any of these
switches through several pages.

1.18 Sampler - Main

The sampler screen consists of a sample-display, a scroll bar to move
around in the sample, and a lot of gadgets.

EXIT

Will close the sampler screen and return the bottom part of the screen
to pattern edit.

STOP

Stops the current sample from playing.

WAVEFORM

Will play the entire sample.

DISPLAY

Will play the visible part of the sample.

RANGE

Will play the selected range of the sample.

CUT

Will cut the selected range of the sample.

COPY

Will copy the selected range of the sample to the copy-buffer.

PASTE

Will paste the selected the copy-buffer to the current position of the cursor.

SHOW RANGE

Will zoom in to the selected range of the sample.

SHOW ALL

Will show the entire sample.

BEG

Will put the cursorline at the beginning of the sample.

END

Will put the cursorline at the end of the sample.

SWAPBUF

Swaps copybuffer with sampledata.

VOLUME

Opens the
volume~requester
.

TUNETONE

Will create a steady sinus tone which you can tune your samples with.

ZOOM OUT

Will zoom out a bit.

RANGE ALL

Select the entire sample as range.

SAMPLE

Will first enter the monitor screen. Now click right button to sample, left to exit. The rate used will be the one specified in the box to the immediate right of the "sample" gadget.

RESAMPLE

Transposes the sample itself. (Changing the pitch) Instructions:
1. Turn on the tuning tone.
2. Use the keyboard to find what note it is. Use finetune if needed.

3. Enter the note in the "Note:" box to the right.
4. Press resample!

DISP

The number of bytes being shown on screen.

POS

The current postition of the cursor.

RANG

The length of the selected range.

INVERT

Turns the sample upside-down. (Flips it around the x-axis!)

MAXIMIZE

Up/down-shift the sample in order to make the best possible effects on a normalize function (see below), and performs it. This will cause maximum volume without distortion.

NORMALIZE

Adjusts the volume of the sample to the highest possible.

NORMALDC

Balances the sample in Y-direction. (Bias adjustment)

BACKWARDS

Turns the sample backwards.

BOOST

Will turn up the treble of the sample. Use this on hi-hats and snares!
If a range is selected, this will only affect the selected range.

INTERPOL

Will Delta-filter the sample. Use this on noisy basses. If the sampler a
range is selected, this will only affect the selected range.

UPSAMPLE

Will remove every second byte of the sample, halving the length and
shifting the pitch one octave up.

DOWNSAMPLE

Will double every byte of the sample, doubling the length, and shifting
the pitch one octave down.

1.19 Sampler - Volume requester

The volume requester works on the current range.

Set the "from" and "to" volume percentages by using the sliders, or just
type in any number you please (from 0 to 200) in the percentage boxes to
the right.

RAMP

Will ramp (calculate) the volume!

NORMALIZE

Will find the highest volume settings possible (without clipping).

@{ub}

Will set the percentages 100%-0%

/

Will set the percentages 0%-100%

-

Will set the percentages 100%-100%

CANCEL

Will exit the volume requester.

1.20 Nice to know

This chapter explains some things about ProTracker which I weren't ↩
sure
of where to put.

Pointer~colors

Keyboard~shortcuts

Effect~commands

Requesters~and~I/O

Standard~paths

Keyboard~percussion

What~is~DYN-samples

1.21 Pointer colors

ProTracker uses the color of the pointer to tell you what's happening.
These colors are:

```
Gray    - Nothing's happening.
Yellow  - Playing song / pattern.
Green   - Disk action.
Blue    - Edit / record.
Magenta - Waiting for input (text, number or something else).
Cyan    - Select entry or delete.
Red     - Something went wrong.
```

1.22 Keyboard

KEYBOARD

The keymap on ProTracker is a standard US keymap. Remember to always use the left shift and alt, as the right ones are used for play / pattern play.

```
    High notekeys:  2 3   5 6 7   9 0   =
                   Q W E R T Y U I O P [ ]

    Low notekeys:    S D   G H J   L ;
                    Z X C V B N M , . /
```

F1 - Choose low octave (C-1 to G-3)
F2 - Choose high octave (C-2 to B-3)

F3 - Cut (sample)
F4 - Copy (sample)
F5 - Paste (sample)

shift+F3 - Cut track to buffer
shift+F4 - Copy track to buffer
shift+F5 - Paste track-buffer to track

alt+F3 - Cut whole pattern to buffer
alt+F4 - Copy whole pattern to buffer
alt+F5 - Paste patt-buffer to pattern

ctrl+F3 - Cut commands to buffer
ctrl+F4 - Copy commands to buffer
ctrl+F5 - Paste cmd-buffer to track

F6 - Go to patternposition 0
F7 - Go to patternposition 16
F8 - Go to patternposition 32
F9 - Go to patternposition 48
F10- Go to patternposition 63

shift+F6-F10 - Store current patternposition on selected F-key
alt+F6-F10 - Play pattern from the stored patternposition
ctrl+F6-F10 - Record from the stored patternposition

Esc - Exit a lots of things.

shift+Return - Insert blank note at cursorpos
and move the rest down.
shift+Backspace - Delete note above cursorpos and move the rest up.

alt+Return - As above, but with all 4 tracks
alt+Backspace - As above, but with all 4 tracks

ctrl+Return - Push cmds one down
ctrl+Backspace - Drag cmds one up

ctrl+0-9 - Select how many steps ProTracker will jump down each time
you insert a note (only in edit-mode)

```
alt+cursor right   - patternnumber up
alt+cursor left    - patternnumber down
shift+cursor right - song-position up
shift+cursor left  - song-position down

ctrl+cursor left   - samplenumber up
ctrl+cursor right  - samplenumber down

Space - Toggle between stop/edit-mode

< (beside Z) - Stop all sound
right Amiga  - Play Pattern
right Alt    - Play Song
right Shift  - Record
Caps Lock    - Toggle Keyrepeat on/off

Del       - Delete note under cursor
alt+Del   - Delete command only
shift+Del - Delete note and command

On Numeric pad:

0 - Select Sample $0
1st row    - Select Sample $1-$4
2nd row    - Select Sample $5-$8
3rd row    - Select Sample $9-$c
4th row    - Select Sample $d-$f
Just Enter - Select Sample $10

Holding Enter + the other keys, will select sample $11-$1F
(Not on A1200, that is!) :(

Period (.) - Kill current sample

Left Amiga (Plus keys below) - Transposing

Sample/Track            Sample/Pattern
---------------------------------
1 - Note Up             2 - Note Up
Q - Note Down           W - Note Down
A - Octave Up           S - Octave Up
Z - Octave Down         X - Octave Down

All/Track           All/Pattern
-------------------------------
3 - Note Up             4 - Note Up
E - Note Down           R - Note Down
D - Octave Up           F - Octave Up
C - Octave Down         V - Octave Down

Tab      - Move cursor to next track
Shft+Tab - Move cursor to prev track

ctrl+A - Toggle channel on/off (+shift = solo channel)
ctrl+B - Mark block
ctrl+C - Copy block to buffer
```

```
ctrl+D - Delete block, drag notes up
ctrl+E - expand track
ctrl+F - toggle filter on/off
ctrl+G - Boost all samples
ctrl+H - Transpose block up
ctrl+I - Insert block, push notes down
ctrl+J - Join-paste block
ctrl+K - Kill to end of track (+shift = to start of track)
ctrl+L - Transpose block down
ctrl+M - Toggle multikeyboard on/off
ctrl+N - Re-mark last block
ctrl+O - Contract track
ctrl+P - Paste block
ctrl+Q - Unmute all channels
ctrl+R - Restore F6-F10 positions
ctrl+S - Toggle split keyboard on/off
ctrl+T - swap tracks
ctrl+U - undo last change
ctrl+V - Filter all samples
ctrl+W - Polyphonize block
ctrl+X - Cut block to buffer
ctrl+Y - Backwards block
ctrl+Z - Restore Effects

shft+0-9 - Store current command on selected key
alt+0-9  - Insert command in current track

alt+"\" - Copy command above cursor to current patternposition.
alt+"=" - Copy command above cursor to current patternposition
          and add one to the value.
alt+"-" - Copy command above cursor to current patternposition
          and subtract one from the value.

alt+A - Monitor/Start sampling
alt+B - Boost sample
alt+C - Toggle channel 3
alt+D - Go to
            disk~operations
             screen
alt+F - Filter sample
alt+I - Toggle AutoinsertEffect on/off
alt+K - Delete current sample/track
alt+M - Toggle metronome on/off
alt+Q - Quit ProTracker
alt+R - Resample
alt+S - Go to
            sampler
             screen
alt+T - Tuning tone
alt+V - Toggle channel 4
alt+X - Toggle channel 2
alt+Y - Save all samples
alt+Z - Toggle channel 1

alt+shift+M - Set metrochannel to current channel

\                  - Set
```

```
              keyboard~percussion
               mode
    Alt+any keypad key - tune percussion note

    Return    - Step one note forward

    Backspace - Step one note backward

    LeftAmiga+N - ScreenToBack
    LeftAmiga+M - ScreenToFront

    When inserting, pasting or join-pasting, hold down shift to keep
    the cursor from jumping to the end of the block. e.g. shft+ctrl+P
```

1.23 Effect commands

```
                EFFECT COMMANDS

Effect commands on ProTracker should be compatible with all other
trackers.

    0 -
              None/arpeggio
                         8 - [Not used]
    1 -
              Slide~pitch~up
                        9 -
              Sample~offset
                  2 -
              Slide~pitch~down
                       A -
              Slide~volume
                  3 -
              Toneportamento
                        B -
              Jump~to~position
                  4 -
              Vibrato
                               C -
              Set~volume
                  5 -
              3~+~A
                                 D -
              Break~pattern
                  6 -
              4~+~A
                                 E - Miscellaneous commands (See below)
    7 -
              Tremolo
                               F -
              Set~speed
              MISCELLANEOUS COMMANDS

The E command has been altered to contain more commands than one.
```

```
    E0 -
                Set~low-pass~filter
                    E8 - [Not used]
    E1 -
                Fineslide~pitch~up
                     E9 -
                Retrig~note
                    E2 -
                Fineslide~pitch~down
                   EA -
                Fineslide~volume~up
                    E3 -
                Glissando~control
                      EB -
                Fineslide~volume~down
                    E4 -
                Vibrato~control
                        EC -
                Cut~note
                    E5 -
                Set~finetune
                           ED -
                Delay~note
                    E6 -
                Pattern~loop
                           EE -
                Delay~pattern
                    E7 -
                Tremolo~control
                        EF -
                Invert~loop
```

1.24 Arpeggio effect command

```
Cmd 0. Arpeggio [Range:$0-$F/$0-$F]

Usage: $0 + 1st halfnote add
          + 2nd halfnote add

Arpeggio is used to simulate chords. This is done by rapidly changing
the pitch between 3(or 2) different notes. It sounds very noisy and
grainy on most samples, but ok on monotone ones.

Example: C-300047  C-major chord: (C+E+G  or C+4+7 halfnotes)
         C-300037  C-minor chord: (C+D#+G or C+3+7 halfnotes)
```

1.25 Slide up effect command

```
Cmd 1. Portamento up [Speed:$00-$FF]

Usage: $1 + portamento speed
```

```
Portamento up will simply slide the sample pitch up. You can NOT slide
higher than B-3! (Period 113)

Example: C-300103  1 is the command, 3 is the portamentospeed.

NOTE: The portamento will be called as many times as the speed of the
song. This means that you'll sometimes have trouble sliding accuratly.
If you change the speed without changing the sliderates, it will sound
bad...
```

1.26 Slide down effect command

```
                Cmd 2. Portamento down [Speed:$00-FF]

Usage: $2 + portamento speed

Just like
                command~1
                , except that this one slides the pitch down instead.
(Adds to the period).

You can NOT slide lower than C-1! (Period 856)

Example: C-300203, where 2 is the command, 3 is the portamentospeed.
```

1.27 Toneportamento effect command

```
Cmd 3. Tone-portamento [Speed:$00-$FF]

Usage: Dest-note + $3 + slidespeed

This command will automatically slide from the old note to the new. You
don't have to worry about which direction to slide, you need only set
the slide speed. To keep on sliding, just select the command $3 + 00.

Example: A-200000  First play a note.
         C-300305  C-3 is the note to slide to, 3 the command,
                   and 5 the speed.
```

1.28 Vibrato effect command

```
                Cmd 4. Vibrato [Rate:$0-$F,Dpth:$0-$F]

Usage: $4 + vibratorate + vibratodepth

Example: C-300481  4 is the command, 8 is the speed of the vibrato,
                   and 1 is the depth of the vibrato.

To keep on vibrating, just select the command $4 + 00. To change the
vibrato, you can alter the rate, depth or both. Use command
```

```
                E4x
                 to
change the vibrato-waveform.
```

1.29 Toneportamento and volumeslide effect command

```
                Cmd 5. ToneP + Volsl [Spd:$0-$F/$0-$F]

Usage: $5 + upspeed + downspeed

This command will continue the current
                toneportamento
                 and

                slide~the~volume
                 at the same time. Compatible with Noisetracker 2.0.

Example: C-300503  3 is the speed to turn the volume down.
         C-300540  4 is the speed to slide it up.
```

1.30 Vibrato and volumeslide effect command

```
                Cmd 6. Vibra + Volsl [Spd:$0-$F/$0-$F]

Usage: $6 + upspeed + downspeed

This command will continue the current
                vibrato
                 and
                slide~the~volume
                 at
the same time. Compatible with Noisetracker 2.0.

Example: C-300605  5 is the speed to turn the volume down.
         C-300640  4 is the speed to slide it up.
```

1.31 Tremolo effect command

```
                Cmd 7. Tremolo [Rate:$0-$F,Dpth:$0-$F]

Usage: $7 + tremolorate + tremolodepth

Tremolo vibrates the volume.

Example: C-300794  7 is the command, 9 is the speed of the tremolo,
                   and 4 is the depth of the tremolo.

To keep on tremoling, just select the command $7 + 00. To change the
tremolo, you can alter the rate, depth or both. Use command
                E7-
```

```
                to
change the tremolo-waveform.
```

1.32 Sample offset effect command

```
Cmd 9. Set SampleOffset [Offs:$00-$FF]

Usage: $9 + Sampleoffset

This command will play from a chosen position in the sample, and not
from the beginning. The two numbers equal the two first numbers in the
length of the sample. Handy for speech samples.

Example: C-300923  Play sample from offset $2300.
```

1.33 Volume slide effect command

```
                Cmd A. Volumeslide [Speed:$0-$F/$0-$F]

Usage: $A + upspeed + downspeed

Example: C-300A05  5 is the speed to turn the volume down.
         C-300A40  4 is the speed to slide it up.

NOTE: The slide will be called as many times as the
                speed~of~the~song
                .
The slower the song, the more the volume will be changed on each note.
```

1.34 Position jump effect command

```
Cmd B. Position-jump [Pos:$00-$7F]

Usage: $B + position to continue at

Example: C-300B01  B is the command, 1 is the position to
                   restart the song at.

This command will also perform a pattern-break (see 2 pages below).

You can use this command instead of restart as on Noisetracker, but you
must enter the position in hex!
```

1.35 Set volume effect command

```
Cmd C. Set volume [Volume:$00-$40]

Usage: $C + new volume

Well, this old familiar command will set the current volume to your own
selected. The highest volume is $40. All volumes are represented in hex.
(Programmers do it in hex, you know!)

Example: C-300C10  C is the command, 10 is the volume (16 decimal).
```

1.36 Pattern break effect command

```
Cmd D. Pattern-break [Pattern-pos:00-63, decimal]

Usage: $D + pattern-position

This command just jumps to the next song-position, and continues play
from the patternposition you specify.

Example: C-300D00  Jump to the next song-position and continue play
from
                   patternposition 00.
     Or: C-300D32  Jump to the next song-position and continue play
from
                   patternposition 32 instead.
```

1.37 Adjust tempo effect command

```
Cmd F. Set speed [Speed:$00-$FF]

Usage: $F + speed

This command will set the speed of the song.

blank: Range 01-FF - Normal timing
  CIA: Range 01-1F - Set vblank speeds
                     with CIA timing.
  CIA: Range 20-FF - Set BPM speeds,
                     range 32-255.
 Both: Range 00    - STOP song.
```

1.38 Set filter miscellaneous effect command

```
Cmd E0. Set filter [Range:$0-$1]

Usage: $E0 + filter-status

This command changes the status of the low-pass filter of your Amiga.
This filter is not present in the A1000 and some very old A500s and
```

```
A200s, but most computers are equipped with it. (If you've got one of
these, please leave this switch alone, since flipping it constantly
garbles the sound.)

Example: C-300E01  disconnects filter (turns power LED off)
         C-300E00  connects filter (turns power LED on)
```

1.39 Fineslide up miscellaneous effect command

```
                Cmd E1. Fineslide up [Range:$0-$F]

Usage: $E1 + value

This command works just like the
                normal~portamento~up
                , except that it
only slides up once. It does not continue sliding during the length of
the note.

Example: C-300E11  Slide up 1 at the beginning of the note.

(Great for creating chorus effects)
```

1.40 Fineslide down miscellaneous effect command

```
                Cmd E2. Fineslide down [Range:$0-$F]

Usage: $E2 + value

This command works just like the
                normal~portamento~down
                , except that it
only slides down once. It does not continue sliding during the length of
the note.

Example: C-300E26  Slide up 6 at the beginning of the note.
```

1.41 Glissando control miscellaneous effect command

```
                Cmd E3. Glissando Ctrl [Range:$0-$1]

Usage: $E3 + Glissando-Status

Glissando must be used with the
                tone-portamento
                 command. When glissando
is activated,
                toneportamento
                 will slide a halfnote at a time, instead of
a straight slide.
```

```
Example: C-300E31  Turn Glissando on.
         C-300E30  Turn Glissando off.
```

1.42 Vibrato control miscellaneous effect command

```
                Cmd E4. Set vibrato waveform [Range:$0-$3]" link "Vibrato effect
command" 0} waveform [Range:$0-$3]

Usage: $E4 +
                vibrato
                -waveform

Example: C-300E40  Set sine(default)
              E44  Don't retrig WF
         C-300E41  Set Ramp Down
              E45  Don't retrig WF
         C-300E42  Set Squarewave
              E46  Don't retrig WF
         C-300E43  Set Random
              E47  Don't retrig WF
```

1.43 Set finetune miscellaneous effect command

```
Cmd E5. Set finetune [Range:$0-$F]

Usage: $E5 + finetune-value

Example: C-300E51  Set finetune to 1.

Use these tables to figure out the finetune-value.

    Finetune: +7 +6 +5 +4 +3 +2 +1  0
       Value:  7  6  5  4  3  2  1  0

    Finetune: -1 -2 -3 -4 -5 -6 -7 -8
       Value:  F  E  D  C  B  A  9  8
```

1.44 Pattern loop miscellaneous effect command

```
Cmd E6. PatternLoop [Loops:$0-$F]

Usage: $E6 + number of loops

This command will loop a part of a pattern.

Example: C-300E60  Set loopstart.
         C-300E63  Jump to loop 3 times before playing on.
```

1.45 Tremolo control miscellaneous effect command

```
                Cmd E7. Set tremolo waveform [Range:$0-$3]" link "Tremolo effect
command" 0} waveform [Range:$0-$3]

Usage: $E7 +
                tremolo
                -waveform

Example: C-300E70  Set sine(default)
              E74  Don't retrig WF
         C-300E71  Set Ramp Down
              E75  Don't retrig WF
         C-300E72  Set Squarewave
              E76  Don't retrig WF
         C-300E73  Set Random
              E77  Don't retrig WF
```

1.46 Retrig note miscellaneous effect command

```
Cmd E9. Retrig note [Value:$0-$F]

Usage: $E9 + Tick to Retrig note at.

This command will retrig the same note before playing the next. Where to
retrig depends on the speed of the song. If you retrig with 1 in speed 6
that note will be trigged 6 times in one note slot. Retrig on hi-hats!

Example: C-300F06  Set speed to 6.
         C-300E93  Retrig at tick 3 out of 6.
```

1.47 Fine volumslide up miscellaneous effect command

```
                Cmd EA. FineVolsl up [Range:$0-$F]

Usage: $EA + value

This command works just like the
                normal~volumeslide
                 up, except that it
only slides up once. It does not continue sliding during the length of
the note.

Example: C-300EA3  Slide volume up 1 at the beginning of the note.
```

1.48 Fine volumslide down miscellaneous effect command

```
                Cmd EB. FineVolsl down [Range:$0-$F]
```

```
Usage: $EB + value

This command works just like the
                normal~volumeslide
                 down, except that it
only slides down once. It does not continue sliding during the length of
the note.

Example: C-300EB6  Slide volume down 6 at the beginning of the note.
```

1.49 Cut note miscellaneous effect command

```
Cmd EC. Cut note [Value:$0-$F]

Usage: $EC + Tick to cut note at.

This command will cut the note at the selected tick, creating extremely
short notes.

Example: C-300F06  Set speed to 6.
         C-300EC3  Cut at tick 3 out of 6.

Note that the note is not really cut, the volume is just turned down.
```

1.50 Delay note miscellaneous effect command

```
Cmd ED. NoteDelay [Value:$0-$F]

Usage: $ED + ticks to delay note.

This command will delay the note to the selected tick.

Example: C-300F06  Set speed to 6.
         C-300ED3  Play note at tick 3 out of 6.

If you use ED0, the note will be delayed a little anyway. You can play
the same note on two channels, delay one, and get a nice flanging
effect.
```

1.51 Delay pattern miscellaneous effect command

```
Cmd EE. PatternDelay [Notes:$0-$F]

Usage: $EE + notes to delay pattern.

This command will delay the pattern the selected numbers of notes.

Example: C-300EE8  Delay pattern 8 notes before playing on.

All other effects are still active when the pattern is being delayed.
```

1.52 Invert loop miscellaneous effect command

```
Cmd EF. Invert Loop [Speed:$0-$F]

Usage: $EF + Invertspeed

This command will need a short loop ($10,20,40,80 etc. bytes) to work.
It will invert the loop byte by byte. Sounds better than funkrepeat...

Example: C-300EF8 Set invspeed to 8.

To turn off the inverting, set invspeed to 0, or press ctrl + Z.

This effect will trash the sample.
```

1.53 Requesters and I/O

```
ARE YOU SURE? REQUESTER

In addition to the gadgets, you can use "Y" or Return for Yes, and "N"
or ESC for No.



THE TEXT-INPUT ROUTINE

Now this is really simple. Clicking on a textline will enable you to
edit it. Some text lines are longer than they seem, so use the arrowkeys
to scroll back and forth in the text. The text input mode is just like
an ordinary text editor. You can use backspace, delete, space and such.
In text-input mode you can also use the numeric pad for entering
numbers. Pressing the right mousebutton will clear the textline and exit
the editing. Use ESC or return to just exit.



THE NUMBERGADGETS

Click on them and type in the value (Hex or Dec). ESC or return aborts.



ARROWGADGETS

Pressing both the left and right button on the arrow-gadgets will speed
them up a bit. All numbergadgets except Finetune and Sample allows you
to click in the gadget and type in the desired value. Holding the right
mousebutton while pressing them will zero the value.
```

1.54 Standard paths

STANDARD PATHS

ProTracker has several standard-paths which it uses when searching for
things. Some of these may be configured, while others can't. These are:

```
        Config-file             "S:"
        Powerpacker-library     "LIBS:"
        Executable modules*     "ST-00:Executable/"
        Modules                 "ST-00:Modules/"
        Songs                   "ST-00:Songs/"
        Instruments*            "ST-00:Instruments/"
        Samples                 "ST-01:"
        Patterns*               "ST-00:Patterns/"
```

*=Not implemented yet.

1.55 Keyboard percussion

KEYBOARD PERCUSSION

Use backspace '\' to toggle modes. One or more dots will appear to the
right of the freemem display.

```
    No dots : Normal keypad.
      1 dot : Drumpad.
     2 dots : Drumpad - Edit/Rec possible.
```

Use Alt + keypad key to set the note for the sample.

With 'drumpad' selected, selecting a sample using the keypad will also
play the sample. This is nice when browsing through the samples of
modules.

1.56 What is DYN samples?

WHAT IS DYN SAMPLES?

DYN samples is a new (and revolutionary) way of playing 16-bit samples
with a 8 - 14 bit replay quality. Your Amiga achieves this by
cheating
a
lot, and this sure does take cpu time. No official replay exist to play
module utilising modules with these samples, so use for own pleasure
only. Maestro 16-bit mono IFF samples are currently the only supported.
:(

The big drawbacks using DYN-samples is that click and ticks may occur.
Also, playing DYN samples at low volumes may cause distortion.

1.57 How to create music using PT

I know. The information in this chapter may not be complete nor easy to understand, but it's atleast far better than any earlier attempt to explain how to use ProTracker in general. Suggestions are welcome!

What~is~a~module?

Patterns

Position-table

Samples

1.58 What is a module?

History

ProTracker's musicfiles are normally called modules. The reason for this is that way back, in the days of SoundTracker v1.0, Karsten Obarski (the author) thought that the best way to save music was as songs with only references to the instrument's names to make it possible to reload them each time the song was reloaded into SoundTracker.

The module format was implemented in SoundTracker v1.0 allright, but it was only intended to be used when programmers wanted to replay the modules in their productions and not as temporary storage, as it is being used today. Because of this, SoundTracker couldn't load modules before version v2.2!

The module-format used today is created by Mahoney and Kaktus which made NoiseTracker in July - August 1989, another tracker-clone. The only big defference they made to the format was that they increased the number of samples from 15 to 31.

What's in a module?

A module basically contains this info:

* Module name
* Samples (name, length, finetune, volume, repeat, replen and the actual audio data)
* Which pattern to play at each position of your module.
* Pattern data

The ProTracker v2.3 also outputs files called "mod!modname" which contains names for each pattern in the module. This will be featured in ProTracker v3.x when the new IFF format is implemented.

A totally empty module (no samples and one pattern) occupies 2128 bytes of disk space, while the (theoretical) largest module one would be able

```
to create would occupy 4326460 bytes! (Even more if
                DYN~samples
                 were
used!)
```

1.59 Patterns

```
                What is patterns?

Any piece of music written with ProTracker is built up from patterns.
Each pattern is built up from four tracks, one for each of the Amiga's
audiochannels. A module may contain up to 64 patterns.

A pattern is 64 steps long. The magnified line is always the one you
edit. If you need shorter patterns, use the

                patternbreak~effect~command
                .

A track is built up of steps like this:

 C-3 01 C20
 (1) (2)(3)

(1) - The first two characters describes the note to play. The "-" may
be replaced by "#" or "b" to describe a flat or a sharp note, depending
on you settings in the
                setup-screen.
                 The third character is the octave
to play the note in. ProTracker supports 3 octaves (1 through 3).

(2) This is the sample-number to play. Valid numbers are hexadecimal 00
through 1F. Sample 0 doesn't really exists but will play the last used
sample for that channel without resetting the volume.

(3) This is the
                effect-command
                 to play.

To sum it up: The C-3 is the note being played. 01 is the samplenumber,
and the three last digits are the effect command, in this case, set
volume to $20 (
                C-Command
                , 20-Value).


Effect tips

You can set the volume to the volume of a specific instrument without
actually trigging the sample.

e.g. "---01000" will set the volume for sample 1 without playing a
note.

Try setting the volume and
```

```
               sliding~the~volume
                down at the same time:
        C-301A08        <--- (If no sample is playing, no sound will be
        ---01A08              heard at all...)
        ---01A08
```

This will create a strange arpeggiato effect.

1.60 Position table

```
               What is the position-table?
```

ProTracker holds a table with information about which order to play the patterns. You may freely define the size of this table. This table may be up to 128 positions, and each position may contain any of your patterns.

When ProTracker plays your module it first play position 0, then position 1 and so on until it reaches the end of your position-table. When this occurs, ProTracker jumps back to position 0. If you want ProTracker to jump to another position, you must use the position jump effect command at the end of the pattern at the last position.

How to edit the position-table

The position-table is controlled using
```
               the~three~gadgets
                labeled "Pos",
```
"Pattern" and "Length". The "Pos"-gadget states which position you are currently editing, while the "Pattern"-gadget states which pattern to play at that position and finally the "Length"-gadgets sets the length of your position-table.

You must remember, though, that the length also implies position 0. This means that if your length is 2, position 0 and 1 will be played. The "I" and "D" gadgets are your friends and are used for inserting and deleting positions.

1.61 Samples

What are samples?

Samples are sound, converted from analog signals to digital (numbers). When your Amiga plays these sounds it converts them back to analog signals and feeds them to your monitor or amplifier. In memory, samples are stored as a big bunch of numbers. The easiest way to think of samples is recorded sound.

What decides the quality of samples? (Hardware approach)

The Amiga's sample-resolution of 8 bit means that it utilises 256 values (2^8) for describing the amplitude of the sound-wave. A CD-player with 16 bits uses thus 65536 values and therefore gives a more accurate reproduction lof the original sound.

Another thing which (heavily) influences the quality of a sample is how often the hardware (in this case, your Amiga) is capable of send out a new value. If you send out a zillion values each second, there's small changes you would miss a change in the sound-wave. Easy? Your Amiga has actually no absolute limits for how fast it is able to reproduce sound, but the easiest output method involves timing the sound output to the Amiga's display hardware. At normal resolutions this means a maximum of ~28000 Hz. In comparison a CD player outputs at a rate of 44100 Hz.

It is generally a common mistake to refer to the Amiga's maximum rate as 28000 samples/second. In order for one hertz (Hz) to occur, you need a positive and a negative value. To put it another way, each hertz consists of two sample values. This means that in the Amiga's maximum sample rate a sample of 56000 bytes would replay in one second!

The sample rate is not constant, as it is used to specify the note to play the sample with. 28000 Hz equals the A-3 note in ProTracker, while the A-2 is half the rate (14000 Hz).

Where to get samples from

Sampling samples yourself is perhaps the easiest (and funniest) way to achieve samples. In order for this to work, you need to buy a sampler (generally very cheap, about £20 should get you a low-end one) and have a source to sample from (expensive). Sampling from CDs is difficult and often there are other instruments in the background, so obtaining all the samples you need from CDs is nearly impossible. The second alternative is buying a keyboard to sample from, but since most of these are pretty expensive (the price is usually proportional to the sound quality) I wouldn't advice anyone to buy a keyboard unless they're pretty sure they want to do music.

Another (well-hated) way to get samples from is by stealing from other modules. It is generally a bad idea to load a module and use clear song in order to use the exact samples as another module. The better approach is to save samples from all modules and mix them together on a HD or create your own sampledisks. This way (if you catalogize them properly) you could easily find the sample you need when you need it.

The third way to obtain samples is obtaining PD sample disks. Your could buy the AM/FM series sampledisks (from Bjørn A Lynne) or you could write to a PD company. (Most of these has a large variety of sample disks.)

What decides the quality of samples? (Software approach)

When sampling, it's essential that your equipment is in perfect order. All connectors must be clean and your wires should be properly isolated. If you experience a lot of interference, you could try to disconnect all electrical appliances nearby.

Sample from the auxillery output of your Stereo rather than from the
headphones output. If your Stereo does not have an AUX output, try
sampling directly from the CD- or Cassette player.

The volume should be adjusted in a way that it just about fills the
entire height in the sample editor in ProTracker in order to utilise the
full sample-resolution.

When selecting sample rates, you should keep in mind that higher
sampling rates yields better quality but effectively eats up a lot of
memory. I myself samples most things at C-3 (16000 Hz) except drums,
which I sample at G-3 or A-3 (28000 Hz).

It doesn't matter (well, mostly) which sampling software you use, but I
prefer to use ProTracker since it's editing functions are faster than
any other I've seen.

1.62 Miscellaneous

Not much to say, these are... erm.. miscellaneous things!

Chronicles~and~anecdotes

Known~bugs

Things~we'll~never~implement

Things~we~will~implement

Frequently~asked~questions

Contacting~the~authors

1.63 Chronicles and anecdotes

This chapter contains some historical and background information. ↩
Not
all of this may be proper to put into a documentation, but I hope it's
interesting for someone.

ProTracker~development~and~progress

ProTracker-relevant~clones

ProTracker~version~history

1.64 Development and progress

SOUNDTRACKER - THE BEGINNING OF A NEW ERA

Way back (dunno, the year) Karsten Obarski was tired of the monotonous music in all games and demos and came up with the idea of using samples for instruments in music. The music used at the time consisted either of synthetic instruments or sampleloops. There were two mainreasons for this. Firstly, people still wanted to support the first A1000 models which came with 256KB RAM and secondly sampleloops were easier to make (mostly grabbed from records).

Karsten Obarski's program, SoundTracker v1.0 didn't look much like todays trackers, it had no sampleeditor, the patterns didn't scroll when playing and one couldn't load modules (storage of music was based on songs). The module/song-format was easy and fairly effective and is still used in today's Trackers (though it has been expanded from 15 to 31 instruments).

The SoundTracker became immediately extremely popular and everyone started using it. The program was far easier to use than other programs available at the time and various "crackers" started improving on the program.

SOUNDTRACKER - THE CHAOS AND THE FALL

As several other groups started making their own versions of SoundTracker, Karsten Obarski gave it up and didn't release more version than v1.0. Groups such as The New Masters, D.O.C. and Spreadpoint seemed to fight for doing the best improvements and this "race" caused a significant improvement of SoundTracker. But after a while the groups got tired of the work and couldn't compete with a new tracker-clone by Mahoney and Kaktus of Northstar called NoiseTracker.

PROTRACKER ENTERS THE STAGE

Just before SoundTracker died, a small and (fairly) unknown norwegian group called Amiga Freelancers started disassembling SoundTracker v2.4 in an attempt to improve it. NoiseTracker gained a lot of users before ProTracker acctually got released. However, ProTracker contained so many features that weren't in SoundTracker, that Mahoney and Kaktus gave up.

ProTracker shortly became the standard and noone even thought of using other trackers.

AMIGA FREELANCERS GIVE UP

I late 1991, Amiga Freelancers got tired of updating the ProTracker and decided to hand out the source to the public. At this time they had recently released

v1.2
and
v2.0
(v2.0 was a semi-commercial version, but the only differences from v1.2 was the opening-graphics). After a while,

they still released
v1.3A~and~v1.3B
with some bugfixes, but the sources
for these were never released.

At this time, a lot of quick hacks was released by several groups in an attempt to win fame and fortune, but the only groups that did noticable improvements was Noxious and Cryptoburners.

NOXIOUS TAKES OVER - FOR A WHILE

Noxious releases their first version (
v2.1A
) in the early 1992 but stops
after
v2.3a
in January 1993 probably as a result of Cryptoburners new
ProTracker,
v3.0
. In my humble opinion Noxious did pretty much for ProTracker and a lot of people still uses v2.3a because they can't get used to the new look introduced by Cryptoburners in v3.0.

CRYPTOBURNERS - RADICAL CHANGES

Cryptoburners changed the resolution of the ProTracker from Low-Res to Hi-Res, a change that has been discussed widely by ProTracker users. They altered the positions of a lot of gadgets and some people still haven't get used to it. Other changes by Cryptoburners included new

disk-operations~screen
and some more
edit-gadgets
.

MAW - 14 BIT SOUND

After ProTracker
v3.10
, Cryptoburners handed the source over to Maw who
was supposed to implement
DYN~sound
(up to 14 bit sound resolution) in
ProTracker. According to Maw himself, no versions after v3.10 was released as these were supposed to be beta-versions sent to different people. Maw never fully completed the DYN implementation and has given up the project and is now working on a new music-program he intend to call ProTracker4. This program has more than 4 voices, synthetic sound and works with 16-bit samples. Maw refuses to release his work on ProTracker and his work is therefore lost forever. However, a lot of people look at this as a nice thing anyway, since it is nearly impossible to play 14-bit sound in an action-paced game and the implementation of the DYN system mostly caused bugs.

CRYPTOBURNERS AGAIN - WHAT HAPPENED TO THEM?

As far as I know, Cryptoburners has stopped developing the ProTracker and is currently working for FunCom productions, a manufacturer of video games. If they still has the v3.10 source it would be the best point to continue the ProTracker from, but the author haven't been able to reach them yet.

1.65 Other tracker-clones

All programs are listed in estimated time of arrival. 8)

SoundTracker v1.0

This is the mother of all tracker-clones and was made a looong time ago (around the release of A500 methinks). The author is Karsten Obarski and the program was (in my opinion) far from easy to use. After releasing this, other groups released a lot of improvements on this program and Karsten Obarski sort of gave up improving the program himself.

SoundTracker v1.1 - v2.3

These versions were different improvements over SoundTracker v1.0. Unfortunately, I have no information on who made the different versions (please write if you do). I do however, know that Dr. Mabuse of DOC and T.I.P of The New Masters released most of these versions.

Games Music Creator

This music program looks a lot like the v2.x-soundtrackers and contains a lot of the same features. However it seems recoded and a bit unfinished. The author is Andreas Tadic.

SoundTracker Pro II

This tracker has a totally different look from the other tracker-clones. It looks like it's supposed to have more features, but is far from finished. Unfortunately, there is no text in the program that states who made this program.

NoiseTracker v1.0 - v2.1

NoiseTracker made by Mahoney and Kaktus of Northstar was the first tracker to implement 31 instruments and a configuration-screen! This tracker became immediately very popular, and was probably the king of the scene until ProTracker came along. The NoiseTracker was based on SoundTracker v2.3

SoundTracker v2.4 - v2.5

These were SoundTrackers which adopted the new features of NoiseTracker, including 31 instruments. According to my knowledge, both these versions were made by MnemoTroN of Spreadpoint.

ProTracker

This is about the place where ProTracker first did it's appearance. Beacuse of the many changes from SoundTracker (which it was based on), there were numerous bugs in the first versions.

NoiseTracker v1.3D

I haven't the foggiest about who made this version of NoiseTracker, but as far as I know, it wasn't Mahoney and Kaktus. This version utilised a special "packed" moduleformat.

SoundTracker v2.6

This SoundTracker implemented something no other tracker has ever used, one-channel patterns. The idea is that for each position you have to specify one pattern for each channel. This idea saves some memory, but is harder to edit and did therefore not "win" the audience. This tracker was also made by MnemoTroN.

StarTrekker v1.0 - v1.3

This tracker was the first to utilise more than 4 voices! It could play a maximum of eight voices and included synthetic sounds. (AM and FM sounds!) Unfortunately, this tracker died silently. StarTrekker was based on NoiseTracker v2.0 and was made by Björn Wesen (aka Exolon of Fairlight).

IceTracker v1.0 - v1.1

This tracker was based on SoundTracker v2.6 and was made by Isepic. It consisted of several improvements and add-ons.

MaxTracker v1.0

This tracker was originally based on ProTracker v1.2, and included the possibility to store samples in fast memory. The readme-file that was distributed with this tracker stated that it could play samples of any length, but couldn't get it to load nor sample samples over 64 KB. The author is Sylvain Marchand.

1.66 Version history

```
                VERSION HISTORY

The version histories are listed by versionnumber, not by releasedate.

There is a version around marked V3.18 BETA with reworked setupscreen
and the dir-selector-bug fixed. Since this version is a true
beta-version (i.e: it is not supposed to be spread) it is not considered
a release nor listed separately here.

All versions of ProTracker is based on lower versions, except (ofcourse)
the V0.89 BETA, which is based on SoundTracker v2.5 by MnemoTroN.

If anyone has version-information for versions not listed here or
versions which has no specific information listed, feel free to

                send~them
                 in.

Sources:
    Source for ProTracker V1.2 and V2.0
    Quick doc for PT V0.89 by Mad Martian
    ProTracker V1.0A readme-file
    ProTracker V1.0A readme-file
    ProTracker V1.3B readme-file
    ProTracker V2.3A readme-file
    ProTracker V3.00ß readme-file
    ProTracker V3.01 readme-file
    ProTracker V3.10 readme-file
    ProTracker V3.14B readme-file

Version releasenotes:

                V0.89~BETA
                          Amiga Freelancers

                V1.0A
                               Amiga Freelancers

                V1.0B
                               Amiga Freelancers

                V1.0C
                               Amiga Freelancers

                V1.1A
                               Amiga Freelancers

                V1.1B
                               Amiga Freelancers

                V1.2
                                Amiga Freelancers

                V1.2C
```

```
                Raul Sobon

V1.2D
                Raul Sobon

V1.2E
                Raul Sobon

V1.2F
                Raul Sobon

V1.3A
                Amiga Freelancers

V1.3B
                Amiga Freelancers

V1.8
                 SCL

V2.0
                 Amiga Freelancers

V2.1A
                Mushroom Studios / Noxious

V2.2A
                Mushroom Studios / Noxious

V2.3A
                Mushroom Studios / Noxious

V3.00~BETA
          Cryptoburners

V3.01
                Cryptoburners

V3.10~BETA
          Cryptoburners

V3.14B
              Cryptoburners

V3.15
                Cryptoburners
```

1.67 V0.89 BETA

```
V0.89 BETA (May 1990) - Exclusively for Crusaders

This is the first known release of ProTracker. It was based on
SoundTracker V2.5 by MnemoTroN and made by The Amiga Freelancers.

* Retrig note, note delay, pattern delay and funk repeat didn't work.
```

* Finetune were introduced.

* Online help. =)

1.68 V1.0A

V1.0A (August 1990) - First public release

Many functions not operational in V0.89 were now implemented.

1.69 V1.0B

V1.0B (September 1990) - Bugfix

* PatternLoop (It always jumped to the start of the pattern, not the loop)

* FineTune AND Glissando-Control were in the same command! (E3)

* Fade Up / Down caused shit and even a few gurus!

* Delete Pos inserted pattern 77 (!) at position 127 in the song, causing it to be longer than 80k!

* Filter All Samples was slow as hell because it tried to filter nonexistant samples.

* If you selected sample 0 from the keypad, it always played the last sample played as sample 0!

* A590 harddisk owners couldn't use the keyboard because of the way Mahoney & Kaktus set up the keyboard interrupt (their NT1.1 routine was used).

* Entering the help screen when in the midi screen fucked up the display.

* Escaping from Mix exited from the whole edit op.!

* CutBeg no longer fucks up the Sample length and Pos.

* The help file was FULL of bugs / misspellings.

1.70 V1.0C

V1.0C (October 1990) - Bugfix

* Error in Finetune-table2 (note B-2)

* Abort Load sometimes caused a guru!

* Toneportamento (cmd 3) didn't work correctly with finetune.

* PLST screen was one pixel off.

* MIDI screen was removed.

1.71 V1.1A

V1.1A (December 1990) - Party release with bugs

* Read input in a more system-friendly way.

* Detached from CLI.

* Loaded IFF samples correctly.

* Used sprites to show loop in sampler.

* Several directory paths in disk operations screen.

* Protracker will now run if you have Kickstart 2.0 installed! (A500, A1000, A2000, A3000 etc...)

* Keyboard buffer. Wow!!! Now you can play really fast, and PT won't miss a single key (...well, not as many as it used to before...).

* The vertical blank interrupt no longer patches the vector itself, but uses the AddIntServer function.

* The playroutine creates it's own CIA or VBLANK interrupt, and it will not be removed if a song is playing when you go to CLI/Workbench.

* Improved "Out of memory" handling (hopefully no more gurus...).

* Only the first 2 bytes of a sample will be zeroed (were the first 4).

* Lots of new keyboard commands/shortcuts using the Alt key.

* Vibrato depth changed to be compatible with NT2.0.

* Funk Repeat changed to Invert Loop (may trash your samples though).

* Play samples from the keypad (dot mode) when pressing backslash.

* Quadrascope. Four channel oscilloscope that displays the samples in real- time, even when playing from keyboard. The good old spectrum analyzer is still there, just click on the scope to toggle.

* Tempo gadget. Default is 125 BPM (normal vblank speed), but that can be changed if you select CIA timing on Setup2. Range is 32-255 BPM, and can also be changed with the 'F' effect command (speed/tempo).

* Repeat and replen will be updated in the samples when you change them.

You no longer have to press a key to hear the new loop.

* Protracker should now be able to show up to 10/16 megs of freemem.

* Error messages when PLST or Config not found.

* Click to enter position, pattern and length with the keyboard.

* Choose RAW or IFF when saving samples.

* Samples can be saved with IFF-loops.

* Change path without reading directory.

* All 30 characters in filename/directory stored (was 24).

* Shows directories, just click to add directory to path.

* Parent directory gadget.

* One preset and several default paths for modules/songs/samples.

* PT will now read an "unlimited" number of directory entries (was 200).

* Two setup-screens.

* Override: Load/save sample from samplepath even if there's a path in the samplename. ST-37:bigbadbass with override will be loaded from DF0: (or whatever) instead of ST-37:. You may want to put all your samples in one big directory on your harddisk...

* NoSamples: Will load a song without asking for the samples.

* BlankZero: This will show the samplenumbers in the pattern in the same way as Noisetracker 1.2 (Zeros are blanked out).

* Show Dirs: If on, directories will be shown in Disk Op.

* Show Publ: If on, PT will show all free Public memory, otherwise just free Chip memory.

* CutToBuff: If on, the part of the sample cut away in the sample editor will be saved in the copybuffer.

* Load Loop: Will load loops from an IFF file when loading a sample, or adding a path in the sampleeditor.

* Slow Mode: Use this toggle if you have a 25 Mhz A3000 where everything is faster. Turning it on might help.

* Set Default paths.

* You can set the maximum number of presets (used to be 2500).

* DMA wait. Use this on 25MHz Amigas (normally 300, use 900 on A3000).

* Set tuning tone (note and volume).

* Select CIA or VBlank timing.

* Set default CIA tempo.

* Spectrum Analyzer/VU-meter copper color editor!

* Use Preset removed, inserted Delete Disk instead.

* You can now save Finetune instead of Volume in the PLST.

* ST-disk number changed to hex (ST-00 --> ST-FF)!

* Adding samples with IFF-loops to the presetlist is possible.

* Proper PLST allocation. Protracker will no longer allocate any memory if there's no PLST.

* ST-disk number changed to HEX. This means you can have up to $FF, or 255 ST-disks (used to be 99).

* Sample Graphing like Audiomaster. With Show Range, Show All, Zoom Out, Range All, etc...

* Play either Waveform, Display or Range.

* Repeat points are shown, and can be dragged around.

* Loop on/off toggle.

* A line shows current position when playing back a sample.

* A sampler just like on NT2.0.

* Resample function w/tuning tone.

* Cut, Copy and Paste functions.

* Cursor to beginning/end.

* Swap current sample with copybuffer.

* Transpose All bug removed.

* Move changed to Copy (use this to copy samples too).

* Upsample changed a tiny bit (allocation).

* Most functions will now work with marked ranges.

1.72 V1.1B

V1.1B (August 1991) - Most known version. (Methinks =)

* Many major bugfixes.

* Changed layout for quadrascope.

* Load gadget no longer hangs the program when trying to load a file
that doesn't exist.

* Better filehandling with error messages.

* Repeat and replen check installed (Rel. A hanged when replen
accidentally got the value zero).

* Repeat and replen were swapped when loading an IFF sample.

* Only RepLen values in IFF samples were added in the preset-editor.

* Tune memory wasn't always updated when editing samples.

* Swap Buffer debugged.

* The arpeggio sounded weird because the pitch wasn't set back at the
right time.

* Shift speeds up the scrolling in the PLST and preset-editor.

1.73 V1.2

V1.2 (July 1991?) - Kickstart V2.0 version

Information could not be obtained.

1.74 V1.2C

V1.2C (July 1991?) - Parallel communication version

Made by Raul Sobon. He did also make V1.2D, V1.2E and V1.2F.

* Made real 8 channels possible through parallel communication between
two Amigas.

1.75 V1.3B

V1.3B (Unknown release date)

V1.3A was also accidentally released, but contained many bugs.

* Save module bugfix.

* Not case sensitive on 'MOD.'

* 'MOD.' bug when switching to and from modules dir.

* Inputhandler bugs fixed.

* Single Step forward and backward works properly.

* The Sample function works with OS2.0

* No hang when a R/W-error occurs in OS2.0

* No crash when loading a non-module file.

* MIDI In works as it should!!!

* PT on Pseudo system screen (?)

* [LEFT Amiga] M and [LEFT Amiga] N works within ProTracker.

* Improved multitasking!

* Screen Adjust (for A3000).

* No slowmode necessary for turbo Amigas. Auto init of DMAWait.

* System requesters ON/OFF function.

* Finds volumes and diskdrives in filerequester.

1.76 V1.8

V1.8 (1994?) - Parallel communication version

Made by Bagitman of SCL.

* Made real 8 channels possible through parallel communication between two Amigas.

1.77 V2.0

V2.0 (Unknown release date)

Mainly the same as V1.2. I don't think anyone knew the difference. =O

1.78 V2.1A

V2.1A (Unknown release date)

Made by Mushroom Studios/Noxious. Contained many Bugs.

* mod. prefix was shown in the file-list if it wasn't lowercase only.

* ReturnfromCLI requester enhanced with quitgadget.

* Gadgetpositions fixed properly.

* Accidental and Diskspace bugs fixed.

* Editskip has now got an indicator.

* DISK OP., PLST and PRESET-ED screens have TOP/BOTTOM gadgets.

* PLST screen has now got Up/Down arrows.

* PT can load&save powerpacked samples/modules.

* PT can load&save tracks/patterns.

* Format disk bug has been fixed (I hope).

* Volume,Repeat,Replen,Pos & Mod is now editable from the keyboard. Just click on the numbergadgets.

* You can now choose an autoinserteffect of your own. (One of the ten effectmacros).

* The bug that completely turned the sound off when stopping the recording has now been fixed.

* You are now able to fastscroll the file- and presetlist to the next startcharacter.

* The preset-ed routines have been corrected.To use your old PLST, simply delete preset #0.

* The metronome has now got an accent on the first beat.

* A metronome keyboard toggle has been included.

* The metronomechannel can easily be changed to the current cursorposition.

* Most of the numbergadgets can now be zero'ed.

* Better patternrefresh when doing keyboard I/O.

* PT now reads the dir after rename/save/delete if the Autodirtoggle is on.

* You can solo a channel with the mouse by holding the right mousebutton while pressing 1,2,3 or 4.

* You can pass an argument from workbench/shell/cli to automatically load a module when PT starts.

* Powerpacker crunching/decrunching with powerpacker.library! PT will automaticly append/remove ".pp" to/from the filename.

* You can save an Icon together with the module.

1.79 V2.2A

V2.2A (Unknown release date)

* Fixed the chip-mem bug from PT2.1A. (It didn't work with chipmem only)

* When Preset-ed screen was shown,you couldn't use the gadgets on the lower part of the main- or sample-screen.

* When autoexitdirtoggle was off,PT didn't change the songname when you loaded a new song/module until you exit from disk op.

* If you resampled a sample with big difference between the tunetone and the resampletone then the new length was incorrectly computed.

* If you ran out of memory when allocating crunchbuffersize then ... software failure! This should work just fine now.

* If you loaded a sample with a '.' in the samplename then PT set the length incorrectly in some cases.

* Save module should work fine now.(another nasty bug is swept away!)

* The Sampler works fine on ks2.0 (This version tests diskactivity before sampling, PT1.3 doesn't!) Source by Matrix/LSD.

* The timer is updated when you play a song while jumping out to Workbench.

* The Loopsprites are working fine with screenadjust.

* Stepplay works fine now. Amiga+Backspace is gone though.

* MIDI in functions are working now, I hope!

* Inputhandler debugged a little.

* ShowFreeDiskGadget debugged (again). Before it showed 2 sectors too much.

* If you pressed the deletefile gadget and then pressed a directoryname in the list then the dir was added to the path but the directory wasn't opened. This is now fixed.

* PrintSong now prints the patternlist with decimal values. It prints all patterns.(Older versions of PT missed to print the highest pattern)

* I have implemented a chord maker.

* I have implemented e Position-Editor.

* On request, I have made ''multitasking'' like in PT1.3.

* ProTracker has it's own screen. When the PT screen is in front of all other screens, PT will wake up.

* The browse function is modified to act like PT1.3. It shows the current devices and disknames.

* If you hold the right mousebutton while pressing the browsegadget a disklist will pop up showing all the device- and disknames from the browsegadget in the filenamewindow. Pressing a name will autoload that directory.

* If you press LeftShift+any key 0-9 a-z while in DiskOp, PLST or PresetEd, PT will move you to the position in the list with a name beginning with the corresponding character.

* If you use MIDI I've added two new keys. A-2 and B-2 on the Synthkeyboard are used as SampleNumber Up/Down! I have also Transposed the MIDInotes input one octave down (because Dolphin has a small keyboard and couldn't access the upper octave,C#3 - B-3).

* SaveExecutable option.

* 9 new toggles.

1.80 V2.3A

V2.3A (January 1993)

* Mountlist in PLST is fixed. Before it asked for disks in all drives even if you didn't have extradrives.

* Helpfile fixed. In the betaversion the jumptable was corrupt.

* Pointer color was green after succesful saving. This one is fixed now.

* In the betaversion you couldn't exit POS-ed with ESC.

* PT shouldn't quit with ESC. This is fixed now. Use Alt+Q (This one has been there all the time) or click in the top-left corner to quit.

* There was a bug in the Entry jump routine (Shift+Desired character) in Disk.op, PLST and PresetED. If you had directories in the current path PT jumped to the first DIRECTORY starting with the desired character instead of searching for the first FILE beginning with that character.

* Some minor bugs fixed as well.

1.81 V3.00 BETA

V3.00 BETA (January 1993) - Hires version

Made by Cryptoburners, and based upon V1.3, and does therefore not contain the features of V2.1A - V2.3A.

* PowerPacker support. Powerpacker.library must be present in the "sys:libs/" directory. Version needed: 35.000 or higher. It is however recommended that you have at least 1MB to use this feature with large modules.

* Protracker rearranged to HIRES 640x256x3.

* Format now works with dfx. Remember to specify which drive to format in the "path" string-gadget.

* Midi has been kicked out (temporarily?).

* "SpectrumAnalyzer" has been kicked out.

* Protracker now works on Kickstart 2.04 and higher (SCS/ECS).

* Memory display now displays all memorytypes.

* Improved input handler. PT will now live peacefully with Commodities. We hope.

* Invented new fileformat based on Interchanged File Format (IFF) chunks. See docs elsewhere on this disk for a discussion of this format.

* Hold record mode; waits for keypress before starting to record.

* New filerequester.

* Screen to back gadgets. You may also use Left Amiga + M/N.

* Standalone playroutine supporting Finetune etc. Not fast though :(

* Some speedups in the SampleEd. It was awfully slow at marking in HIRES!

* Scroll bars added here and there (but not everywhere ;)

* New pointer! The old one was simply too big for the new resolution.. :)

* Setupscreen revised and merged into one.

* Date and time online at mainscreen.

* Task interferance problems fixed (Improved multitasking). No busy waits.

* The dragging routine in the sampler draws pixels instead of lines when dragging.

* Major rearranging on mainscreen; editor centered and some new buttons
for editing, such as cut/paste, up/down octave etc.

* You may now click into the different tracks with the left mousebutton!
:o

* Audio.device channel allocation, to ensure that other programs doesn't
mess with your tunes in a multitasking environment.

* Chords editor with up to 7 notes. (Hi JanneS :)

* PT now hopefully works on KS3.0 in AGA mode too (Hi Vishnu).

* The sampler works again! ;)

1.82 V3.01

V3.01 (Unknown release date) - AM/FM release

* Reads the devices mounted in the system instead of hardcoded ones.

* "Free" gadget fixed.

* Sample read/write fixed.

* Now gets the directory automagically when one is entered in the
pathgadget.

* Fixed bug in chord editor. Sometimes caused division by zero.

* Added adjust to chord editor; boosts the volume of the resulting
sample to fit the original sample.

* Screen adjust gadget is now working. :)

* Bug in runback fixed.

* CIA allocation failures are now displayed.

* Tools gadget removed.

* Short keys for setup-items caused the switches to be displayed on the
main screen. Not anymore :)

* Drumpad likewise.

* Several smaller bugs fixed.

* Added a gadget on the dirscreen for fast browsing through the file
type options.

* New icon on the out-of-memory requesters :o

* Some new gadgets are working!

* Minor graphical changes

* Decimal numbers on samplelength and loops

1.83 V3.10 BETA

V3.10 BETA (May 1993) - Amiga Format version

* A new improved 7-note chord editor, recalculates loops.

* Full compatibility with all existing OS versions including OS3.0

* Better multitasking capabilities.

* New sample edit options, includes Invert, Maximize, NormalDC,
Normalize.

* Mouse controlled editing made possible.

* Added real VU-meters.

* New configurable options:
        - Screen positioning.
        - Lace on/off for genlock users.
        - Multicached disk directories.

1.84 V3.14B

V3.14B (Unknown release date) - DYN samples version

* For the first time in Amiga history: Improved sound quality! Actual 14
bit sound dynamics using a new and revolutionary algorithm. No
additional add-ons required, furthermore, no extra storage usage. Fully
compatible with traditional 8-bit Amiga sounds.

* Reads and converts Maestro MONO 16-bit samples.

* Info requester when reading IFF samples.

* The "BlankZero" now also blanks empty effect commands and the extended
command byte.

1.85 V3.15

V3.15 (Unknown release date) - Bugfix version

1.86 Known bugs

```
                If your copy of ProTracker shows some strange behaviour and you ↩
                   can't
find it here, look in the
                Frequently~Asked~Questions~section
                . It could
be that it's not really a bug. If you can't find your case anywhere,
please do
                contact~the~authors
                .

* On A1200/A600 it is not possible selecting (or atleast very difficult)
to
                select~instrument
                 $10-$1F using the keypad. This is our punishment
for not using the system and may not be fixed for a while. 8(

* ProTracker does not go on well with the debugging tool MungWall from
Commodore. Temporary workaround is to quit MungWall before running
ProTracker.

* Very few screenblankers cooperate with ProTracker. If ProTracker locks
up after 2-3 minutes, this could be it. Disable all screenblankers
before running ProTracker. (SuperDark does work though!)

* If you are using a hires-pointer on your Workbench, some gadgets, the
pointer and the cursor will appear with half width.

* On really fast machines (68030 30MHz or faster) some notes causes crap
in the audiospeakers. This will (hopefully) soon be fixed.

* When entering subdirectories in
                disk~operations,
                 some lines remain
"selected". This is taken care of.

* Earlier, you could change instrument-number without setting new notes,
and chip-instruments (very short samples with loop) would automatically
jump to the new sample. This no longer works after the implementation of

                DYN~samples
                .

* Playing the same note on two different channels may produce slightly
different sound from time to time.

* When editing the modulename from
                disk-operations
                , the first character
of the name of instrument 1 is set to a "_".

*
                Effect-command~9
                 (sample offset) doesn't give any sound if the desired
offset is behind the end of the sample. Earlier, it did.
```

```
* If ProTracker attempts to play a sample with 0 in
                replen
                 (this is the
case of some older modules) it will lock up.

* Memory fragmentation occurs when
                starting~PT
                 twice from Icon.

* Samples succeeding 32Kb are not displayed fully in the
                sample~editor
                .

* Keyrepeat sometimes don't stop in pattern editor. (Nasty!)

* Saving samples sometimes chooses the wrong device to write to.

* Turning off audio channels doesn't stop repeating instruments
immediately.

* Updating sample length when resampling/upsampling/downsampling.

* At present time you must
                launch~PT
                 as a project icon from WorkBench,
or from CLI. Tool icons will _hang_ your system!

* Applications that do not allocate the timers properly, will cause the
music to stop (eg. ProPlay). However, this is _not_ ProTrackers fault;
PT allocates the used timers gently :) As you see this is not really a
bug, but we just want kill potential "bug reports" in birth ;)

* When a
                DYN-sample
                 is played by keys while song is playing, it will
sound distorted.

* If starting a song, notes with
                DYN-samples
                 in the first line are not
always played correctly.

*
                Sample~edit~operations
                 on
                DYN-samples
                 will not work, but crash the
system (sooner or later).

* Using volume on
                DYN-samples
                 creates awful distorted sounds.

* When loading a new module, ProTracker doesn't set the lo-pass filter
to the default value.
```

* When clicking on the destination-gadget in the chords-edit, PT seems
to select a random sample-number. This is ofcourse wrong.

* When mixing samples, PT does not calculate the arpeggio-values to put
in the samplename correctly if these are too high.

* Writing values directly to sample length/repeat/replen does not always
work.

* Increasing sample-length using the up arrow is okay, but decreasing
does not seem to work as desired.

* The playtime-clock is halted when the Workbench screen is up.

1.87 Things we'll never implement

* Splitting up each track of a pattern into a seperate "pattern",
independent of the others. After having talked to many musicians, we all
agreed this is all in vain. Nobody seems to be going to use it, due to
the serious timing problems that will arise. Use SIDMON instead, if
you're really anxious to use this feature ;)

* 8 channels Protracker. The Amiga is designed for 4 voices, if this
feature is to be implemented, the CPU has to mix the channels realtime,
and will use all CPU DMA on lowend Amigas. Forget it. Use StarTrekker if
you want it.

* Intuition Protracker (using the system for creating the
user-interface). Too much work :) maybe some sunny day...

1.88 Things we will implement

* Implementation of the new IFF fileformat :) which includes:
  - Commands ranging from 0..0xFF (0..255). All commands that CAN run
    in parallel will do. Lotsa other commands (hi JanneS :).
  - 256 Positions.
  - 256 Patterns.
  - 256 Samples. (Higly wanted!)
  - Implement octave 0!
  - Maximum sample size from 64Kb to 128Kb. It's there, why not use it?

* Digital filters and noise-reduction in the Sampler. Planned filters
include Lowpass, Highpass and Bandpass (FIR filters). Perhaps Graphic
Equalizer too.

* Seek zero/loop in sampler.

* Protracker.library for non-tech programmers wanting to use PT modules
in their programs.

* Separate position-editor.

* Save module as executable.

* Add icon to saved files.

* Load samples/instruments/patterns from an existing module.

* Freehand sample-editing.

* ASDR editor for channel modulation.

* Gain/Finished info when (Power)Packing.

* Hypertext on-line help (a la Windows).

* Serial comm. to other protrackers. (Synchronize playing)

* A whole new range of DSP effects for your samples, including:
  - Low, high, band and allpass filters (IIR filters).
  - Comb filters.
  - Shelving low and high filters.
  - Multitap delay/feedback algorithms.

* Bugfree functionality. (This one has low priority!) =)

* Read (and save?) QuadraComposer (EMOD) modules.

1.89 Frequently asked questions

If you have any questions that's not listed here, send~them in.

Q: Keyboard~shortcuts with more than one key involved doesn't work as expected. I have an A1200 or A600.

A: This is a known bug, and it's not much to do about it.

Q: When I run the program xxx before starting ProTracker, my Amiga crashes.

A: Don't run xxx.

Q: When I launch ProTracker it always locks up after x minutes.

A: You have a screenblanker running. Quit it before launching ProTracker.

Q: I use the resample-function in the

sampler
, but the resulting sample
is one or two octaves too high!

A: Your
tunetone~setting
is not C-3. ProTracker takes the tunetone
setting into account when resampling.

Q: Some of my chipmodules sounded strange on my new ProTracker.

A: Some chip-modules rely on ProTracker changing the sample when setting a new samplenumber for a channel without specifying a new note. This worked earlier, but doesn't any longer.

Q: Sometimes instrument 1 gets the first character of it's named replaced with a "_". Why?

A: This happens when editing the modulename from
disk~operations
and is
a bug. Sorry.

Q: I've got a module which my player regocnises as a ProTracker module, but ProTracker refuses to load it in (or crashes). Why?

A1: The module utilises more than 64 patterns. These modules are created with ProTracker V2.1 - 2.3 by Noxious. It's not much to do about this until we've included support for these modules.

A2: The module is severly damaged. Try to repair it using the TrackerTool command from DOS.

A3: The module is packed with some bogus packer that ProTracker doesn't support. See the documentation for the player to find out which packer and get the packer to unpack the module.

Q: I've got a module which makes the sample-display look phoney and distorts the length-display in the sample-editor. Why?

A: Your module has samples longer than 65536 bytes. ProTracker does not support this at current.

Q: I've got a module which gives a very high peeping sound at the end of some samples. Why?

A: This module is either made on another platform (PC, Macintosh) or converted from some old module-formats. The trick is that ProTracker needs the 2 first bytes of each sample to be set to 0. This could either be done by zooming in, selecting the first 2 bytes and setting their volume to 0 or by cutting one byte at the end of the sample (this last action causes ProTracker to automatically reset the first two bytes).

```
Q: I've got a module which makes ProTracker lock up when I attempt to
play it. Why?

A: Your module probably contains samples with
                 replen
                  set to 0. Set all
of these to 2 and your module should be fine.
```

1.90 Thanks and acknowledgements

```
The authors would like to thank and bring their acknowledgements out
to:

Lizard, all the guys at #amiga and #amigascne.
```

1.91 Contacting the authors

```
Contact one of the authors if you have bugreports or any other
suggestions.

NOTE: There is no use contacting Cryptoburners for anything regarding
ProTracker anymore since they have stopped developing it. I am therefore
collecting all reports until someone continues the ProTracker or I can
convince Cryptoburners to give me the source.

E-Mail: (Preferably)
    havardp@mail.stud.ingok.hitos.no

IRC: (Nice alternative)
    Nick: Howard or Howard_ or Howard__
    Channels: #amiga, #amigascne and #norge.

Snailmail: (When all else fails...)
    Håvard Pedersen
    Kvartsveien 175
    N-9022  KROKELVDALEN
    Norway

Phone: (Voice rate)
    +(47) 77 63 13 34 (Håvard)
```

1.92 For the technical minded

```
                This chapter describes support files and other things needed to ↩
                   know
when writing applications that handles ProTracker files.
```

File formats

ProTracker~song/module~format

ProTracker~IFF~song/module~format
Sources

Using~the~ProTracker~play-routine

Using~the~ProPruner~play-routine

Using~the~SoundFX~sound-replay~engine

Using~the~PTCalcTime~playtime~calculator
Support software

The~TrackerTool~command

ProPacker~v2.1~by~Christian~Estrup

PPRStrip~and~PPRSampler
Miscellaneous

CIA~tempo~calculation

DYN-Samples

1.93 ProTracker song&module format

PROTRACKER SONG/MODULE-FORMAT

Offset	Bytes	Description
0	20	Songname. Remember to put trailing null bytes at the end... Please do also remember that a module may have a 20-char name, which means that the string is not zero-terminated.
20	30	Sample information for sample 1 (See below)
50	30	Sample information for sample 2
80	30	Sample information for sample 3
:	:	
V	V	
890	30	Sample information for sample 30
920	30	Sample information for sample 31
950	1	Songlength. Range is 1-128.

951	1	Well... this little byte here is set to 127, so that old trackers will search through all patterns when loading. Noisetracker uses this byte for restart, but we don't.
952	128	Song positions 0-127. Each hold a number from 0-63 that tells ProTracker what pattern to play at that position.
1080	4	Identifier data.

Pro/Noise/Sound-Tracker	"M.K."
100 pattern ProTracker v2.x	"M!K!"
StarTrekker 4-voice	"FLT4"
StarTrekker 8-voice (not compatible!)	"FLT8"
FastTracker 4-voice (maybe compatible)	"CHN4"
FastTracker 6-voice (not compatible!)	"CHN6"
FastTracker 8-voice (not compatible!)	"CHN8"
ProRunner processed (not compatible!)	"SNT."
Wanton packer (not compatible!)	"WN"+0+crap

I've read somewhere that NoiseTracker modules has the ID "M&K!", but this is not correct.

If the identifier is not there, the text has maybe been removed to make the module harder to rip.

1084	1024	Data for pattern 0. he number of patterns stored is
:	:	equal to the highest patternnumber in the song position
V	V	table (offset 952-1079).
???	???	
???	???	If the file is a module, the sampledata follows right after the patterndata. If not, this is the end. The samples are located from 1 through 31. Use the sampleinfo to find the start and end of each sample.

Each
DYN~sample
 is followed by a longword with length
of the DYN info, and then the DYN info itself. The format of the DYN info is not publically available yet.

SAMPLE INFORMATION

Offset	Bytes	Description
0	22	Samplename. Pad with null bytes. See notes on modulename.
22	2	Samplelength. Stored as number of words. Multiply by two to get real sample length in bytes.
24	1	Lower four bits are the finetune value, stored as a signed four bit number. The upper nibble is set to a 1 if this sample is a

```
                DYN~sample
                .
                Value:  Finetune:
                 0       0
                 1      +1
                 2      +2
                 3      +3
                 4      +4
                 5      +5
                 6      +6
                 7      +7
                 8      -8
                 9      -7
                 A      -6
                 B      -5
                 C      -4
                 D      -3
                 E      -2
                 F      -1
```

25	1	Volume. Range is $00-$40, or 0-64 decimal.
26	2	Repeat point. Stored as number of words offset from start of sample. Multiply by two to get offset in bytes.
28	2	Repeat length. Stored as number of words in loop. Multiply by two to get replen in bytes.

PATTERN FORMAT

Each note is stored as 4 bytes, and all four notes at each position in the pattern are stored after each other.

```
    00 -  chan1  chan2  chan3  chan4
    01 -  chan1  chan2  chan3  chan4
    02 -  chan1  chan2  chan3  chan4
    etc.
```

Info for each note:

```
 BIT#  31  30  29  28  27  26  25  24     23  22  21  20  19  18  17 16
 USE   S7  S6  S5  S4  P11 P10 P9  P8     P7  P6  P5  P4  P3  P2  P1 P0

 BIT#  15  14  13  12  11  10  09  08     07  06  05  04  03  02  01 00
 USE   S3  S2  S1  S0  E11 E10 E9  E8     E7  E6  E5  E4  E3  E2  E1 E0
```

```
    S7-S0  - Samplenumber.
    P11-P0 - Period of note.
    E11-E0 - Effect command.
```

Periodtable for Tuning 0, Normal

```
    C-1 to B-1 : 856,808,762,720,678,640,604,570,538,508,480,453
    C-2 to B-2 : 428,404,381,360,339,320,302,285,269,254,240,226
```

```
    C-3 to B-3 : 214,202,190,180,170,160,151,143,135,127,120,113

To determine what note to show, scan through the table until you find
the same period as the one stored in byte 1-2. Use the index to look up
in a notenames table. These periods are really for NTSC timing, but is
used for both PAL/NTSC.



This is the data stored in a normal module/song. A packed song starts
with the four letters "PACK", and then comes the packed data.
```

1.94 ProTracker IFF song&modules

```
                Introduction

The text below was intended to be the documentation on the fileformat
used in ProTracker. However, we decided to wait with the actual
implementation of the format until having released a couple of versions,
because we'd like to hear some comments, suggestions etc. upon it first.
So read it lightly, and feel free to post your opinion to one of the

                authors
                . Note that since this is only a suggestion, don't start
programming a revolutionary new piece of code based on this info yet; we
may change the format :)... Here we go...

Fields marked "*Reserved*" are reserved for future use and are
guaranteed to cause hangup if messed with.


General

With this release of Protracker we have decided to change the
filestructure of the musicfiles produced with the program. We felt the
old format was too obsolete, messy and out of date for us to use any
further. So we invented this new format. The format is based upon
Interchanged File Format (IFF) chunks, originally developed by
Electronic Arts, but now in widely use on the Amiga. The format allows
considerable flexibility and does not suffer too severly from changes
and updates, and is therefore perfect for our use.


The Format

We will in this section introduce and describe each chunk type appearing
in a Protracker music file. Look in the next section for the sequencial
description.

** Contents of Chunk "VERS":

  OFFSET  Length  Contents  Meaning
  -------------------------------------------------
   0       4      "VERS"    Chunk identifier.
   4       4      ????????  Chunk length (in bytes).
```

8	2	????????	Version number (word).
10	6	"PT3.00"	Version ID string.

This chunk is used by Protracker to identify the producer of the module, and if necessary perform upgrade-conversion if the file was made with a pre- vious version of Protracker. There can be at maximum one "VERS" chunk in a Protracker music file. This chunk is not critical; it may be obmitted, but be aware of the possible incompatibility problems that may arise if it's left out.

** Contents of Chunk "INFO":

OFFSET	Length	Contents	Meaning
0	4	"INFO"	Chunk identifier.
4	4	????????	Chunk length (in bytes).
8	32	[..??..]	Song name (string).
40	2	????????	Number of instruments (word).
42	2	????????	Number of positions (word).
44	2	????????	Number of patterns (word).
46	2	????????	Overall volume factor (word).
48	2	0006h	Default speed (#VB) (word).
50	2	????????	Packed field. See below.

Protracker uses this chunk to set different internal variables, and to store vital information used in replay and processing of the file. The song name is a maximum 32 Chars long ASCII string. It need not be NULL-terminated. Number of instruments indicates the number of instruments used in the song, it may range from 0 to 65535. At present version number, however, there may be maximum 255 instruments in one song. Number of positions reflects the actual length of the song, that is; how many patterns that will be played during a complete cycle. This number may vary from 0 to 65535. Number of patterns, on the other side, reflects how many _different_ patterns that will be played during the song. This number is used to calculate the total length (in bytes) of the song. The Overall Volume factor is used to compute the final volume of all channels after the individual channel-volumes have been figured out. In this way it is easy to control the loudness of the music from the program/ song itself. Default speed is the number of VBlank frames between each pattern position change, and is as default set to 0006h. The packed field consists of these bits (right to left order):

Bit	Meaning	0	1	Def
0	Filter flag.	Filter off.	Filter on.	0
1	Timing method.	VBlank.	CIA timing (BPM).	0
2	File type.	Module.	Song (no instruments).	0
3	Packstatus.	Packed patterns.	Raw patterns.	1
4	Length flag.	Equal pattern length.	Variable pattern length.	0
5	Voices flag.	4 voices.	8 voices.	0
6	Sample res.	8 bit.	16 bit.	0
7	*Reserved*			x
8	*Reserved*			x
9	*Reserved*			x
10	*Reserved*			x

```
  11  *Reserved*                                                      x
  12  *Reserved*                                                      x
  13  *Reserved*                                                      x
  14  *Reserved*                                                      x
  15  *Reserved*                                                      x
  ------------------------------------------------------------------------
```

There can be at most one "INFO" chunk in a Protracker musicfile. This chunk is vital; it _must_ be present for the replay routine to function properly.

** Contents of Chunk "INST":

```
  OFFSET  Length  Contents  Meaning
  ---------------------------------------------------------
   0       4      "INST"    Chunk identifier.
   4       4      ????????  Chunk length (in bytes).
   8      32      [..??..]  Instrument name (string).
  40       2      ????????  Length of instrument (word).
  42       2      ????????  Instrument loop start (word).
  44       2      ????????  Instrument loop length (word).
  46       2      ????????  Instrument volume (word).
  48       2      ????????  Instrument finetuning (integer).
  ---------------------------------------------------------
```

The "INST" chunk is used to store information about an instruments properties, such as length and volume. The instrument name is a maximum 32 Chars long ASCII string. It need not be NULL-terminated. The Length field describes the length of the instrument (in words) and thus ranges from 0 to 128Kb (65535 words). Instrument Loop Start sets the offset from which to start playing after the first replay. This value may vary from 0 to the instrument length. Instrument Loop End sets the length of the loop to play after the first replay, relative to the loop start value. It may thus vary from 0 to [Ins_len-Loop_start]. Instrument volume indicates which volume to use in the replay of the sample, if the song doesn’t say differently. This value varies between 0 and 40h. Instrument finetuning sets the sample-rate correction difference and varies from -7 to 7 (0fff9 to 0007h).

There may be any number of "INST" chunks in a Protracker music file, limited to the number of instruments actually used in the song. This chunk is not vital; it may be left out if the song-only bit of the control word in the "INFO" chunk is set. Otherwise, it should result in an error.

** Contents of Chunk "PPOS":

```
  OFFSET  Length  Contents  Meaning
  -------------------------------------------------
   0        4     "PPOS"    Chunk identifier.
   4        4     0ffh      Chunk length (in bytes).
   8      256     [..??..]  Pattern position table.
  -------------------------------------------------
```

This chunk contains the table defining which pattern to play in a given song- position. Each entry in the table is a byte indicating which out of 256 possible patterns to play. There may be at maximum one "PPOS"

chunk in a Protracker musicfile. This chunk is vital; it _must_ be present to play the song.

** Contents of Chunk "PTRN":

OFFSET	Length	Contents	Meaning
0	4	"PTRN"	Chunk identifier.
4	4	????????	Chunk length (in bytes).
8	32	[..??..]	Pattern name.
40	?	[..??..]	Pattern data.

This chunk is used in a module of variable pattern length. The chunk must thus appear as many times as there are patterns in the song. The chunk length divided by 8 (>>3) will show the pattern length (default 64). Pattern name is a 32 byte long ASCII string, describing the pattern, eg. "Intro part 3". It need not be NULL-terminated. This chunk is critical; it must be present in the file, or it will be regarded invalid. NOTE: This chunk is not in use in the present version (3.00B), and will be ignored if found.

** Contents of Chunk "SMPL":

OFFSET	Length	Contents	Meaning
0	4	"SMPL"	Chunk identifier.
4	4	????????	Chunk length. (In bytes!)
8	?	[..??..]	Raw sample data.

The "SMPL" chunk contains the raw sample data of an instrument. This chunk is not critical; if the song-only bit of the "INFO" chunk is set, it may be obmitted. If, however, the file is a module, then the number of "SMPL" chunks in the file must be equal to or greater than the number of instruments used in the song. If not, the file will be regarded incomplete.

** Contents of Chunk "CMNT":

OFFSET	Length	Contents	Meaning
0	4	"CMNT"	Chunk identifier.
4	4	????????	Chunk length (in bytes).
8	?	[..??..]	Raw ASCII text.

The "CMNT" chunk is used for a signature, comments, greetings, date of completion or whatever information the composer wishes to include with his or hers creation. This chunks is not critical; it may be left out and will typically be ignored by most applications.

These are the chunks that may appear in a Protracker musicfile. If other chunks are encountered, they will be ignored. Any program dealing with this fileformat should perform tests to determine the validity of the file in consideration. Using the Protracker.library will guarantee correct handling of musicfiles, and we strongly encourage the use of

this runtime shared library instead of hacking away on your own. Look elsewhere on this disk for the library documentation, the library can be found in the "LIBS/" directory.

The sequential format ----------

In this section we will describe how the various chunks are expected to be located within the file. These rules _must_ be followed or it will wreak havoc when tried manipulated with inside Protracker. Here comes the header in table form:

OFFSET	Length	Contents	Meaning
0	4	"FORM"	Indicate start of IFF file.
4	4	????????	File length.
8	4	"MODL"	IFF type identifier.

This header must be found in the start of the file, or it will be rejected as not being a Protracker musicfile. From offset 12 in the file, things may vary somewhat. The only rules are these: After a "INST" chunk a "SMPL" or a new "INST" chunk _must_ follow. This "SMPL" chunk will be regarded as the sample data of the instrument(s) preceding it. If after a "INST" chunk another "INST" chunk follows, and the module-flag in the "INFO" chunk is set, then all "INST" chunks following each other will share the same sampledata found in the first "SMPL" chunk after them. Also, all "INST" and "SMPL" chunks must be found in sequence. That is, when a "INST" chunk is found for the first time in a file, all other "INST" and "SMPL" chunks must follow. If this is not so, an error message should be given, and processing terminated. Note that in a song-only file, no "SMPL" chunks should be included. If any "SMPL" chunks are encountered in such a file, they should be ignored and a warning given. All other chunks used in a musicfile may be located anywhere in the file, usually in the beginning of it, but no assumptions of their locations should be taken. Note that all used chunks _must_ be found _before_ the "BODY" chunk, which is the last chunk to be found in the file. Searching for chunks should stop when encountering a "BODY" chunk. The "BODY" chunk is constructed like this:

OFFSET	Length	Contents	Meaning
0	4	"BODY"	Chunk identifier.
4	4	????????	Chunk length (in bytes).
8	?	[..??..]	Raw pattern data.

Chunk summary

Now follows a list of the chunks that have meaning in a Protracker musicfile:

Chunk	Function	Critical?
VERS	Contains information about the producer of the file.	No
INFO	Contains vital information and standard settings.	Yes
INST	Information about instruments: length, volume etc.	Yes

```
 SMPL  Raw sample data associated with one or more instruments.   No
 PPOS  Position table. Information about patternsequence.        Yes
 CMNT  Comments, greetings etc. in ASCII code.                    No
 PTRN  Pattern data for modules with varying patternlengths.     Yes
 BODY  Pattern data for modules with equal patternlengths (def). Yes
 ---------------------------------------------------------------------
```

1.95 Using the PT playroutine

Normal usage

The replay relies on you not turning on the audio DMA, as this is done when needed.

In order to get the replay working properly it needs to initialize itself. To make this happen, call PT_Init with a pointer to your module in A0.

When this is done, everything is ready to go. Simply call PT_Music 50 times a second to achieve music. This call assumes exclusive access to an OCS/ECS/AGA-compatible audio hardware located at $DFF000. It is safe to call both PT_Init and PT_End if your PT_Music-calls are done in an interrupt and your interrrupt is still enabled.

When you're finished, you may call PT_End to ensure that all pending sound is removed.

Mastervolume/balance

In order for the mastervolume features of the replay to be enabled you must insert "PT__MasterVol = 1" somewhere in your source before the replay. You should, however, note that enabling the mastervolume features heavily slows down the replay.

In order to adjust the mastervolume, you may call PT_SetMasterVol with the volume (a word) for the left channel in D0 (range 0-64) and the volume for the right channel in D1. The change takes place immediately.

Overstep

This is a strange word, which I generated myself when I couldn't find another word to describe this feature. The idea behind overstep is that the programmer should, in some way, be able to sense when the current module is finished in order to be able to load in another module or perform some other action based on this.

In order to activate overstep, insert "PT__OverStep = 1" in your sourcecode. Overstep provides nearly no overhead.

The byte PT_OverStep will be written with a 1 if a position jump occured, the module was halted, the end was reached or a fast forward hit the end. The value -1 may occur in this byte if a rewind was done in position 1.

Please note that the replay itself never clears the byte PT_OverStep, so the programmer must do this himself when changes are sensed.

Winding

In order for the winding functions to be enabled, insert "PT__WindFuncs = 1" in your source. You may now call the functions PT_Forward and PT_Rewind to skip forward and backward one pattern at a time. The module loops correctly both at beginning and end.

Please beware that if there are any position jump commands at the end of the current pattern when you do a PT_Forward or a position jump command just occured when doing a PT_Rewind, the positions may not be played in a correct order. Working around this is way to much work, so settle with it.

CIA timing

The replay does not do CIA timing itself, but has been extended with a small support forl the user to implement this. To enable the CIA-mode of the replay, insert "PT__CIA = 1" in your source.

When CIA is enabled, the replay assumes the user to supply a function called PT_SetIntRate wich is to be called with the CIA tempo in D0 and does not trash any registers. This function is called inside PT_Init and each time the module attempts to change the CIA tempo.

68020+ optimizations

To enable the 68020+ optimizations, insert "PT__68020 = 1" in your source. The replay now uses a few 68020+ only instructions to gain some speed. These optimizations aren’t much for now, but may increase in the future.

NoiseTracker compatibility

NoiseTracker is another tracker written by Mahoney and Kaktus which writes modules almost identical to the ones written by ProTracker. As of revision 2C, the ProTracker replay is capable of playing these modules with the extra ’quirks’ of the NoiseTracker module-format. The NoiseTracker compatibility is on by default and may be disabled by inserting "PT__NTComp = 0" in your source.

The NoiseTracker compatibility causes the replay to correctly play NoiseTracker vibrato and utilises the repeat-feature of NoiseTracker. This option uses nearly no (if any) extra CPU-time.

DMA delay (advanced users)

The well-known (and well-cursed) DMA wait is a small pause needed every time the Amiga’s audio hardware is supposed to change from one sample to

another. Commodore states that this delay should be equal to one raster-line.

The ProTracker replay supports selecting this delay in order to save some CPU-usage. The default delay is 80, which fits to Commodore's recommendation. OctaMed by Teijo Kinnunen uses a delay equal to 64, and I guess he haven't had any problems with it. You are not allowed to specify a delay larger than 128.

The set the delay, insert the line "PT__DMADelay = n" in your source, where n is the DMA delay you want. Please remember that this delay does NOT need to be larger on faster machines, as it's timed using the beam-position of the Amiga.

1.96 Using the PPR playroutine

BACKGROUND

The ProPruner playroutine is an attempt to make a replay that is significantly faster than the clean ProTracker replay. The ProPruner playroutine contains nearly all features that one can think of and is pretty fast. In order to achieve the extra speed, it needs the modules to be modified and saved in a special format. Currently, the ProPruner replay need

ProPacker~v2.1

modules and does some internal changes to these modules before playing them. The

ProPacker~v2.1

is made by Christian Estrup and is spread without his permission. For documentation on how to use the ProPacker, please read it's

separate~chapter

. The program

TrackerTool

will soon be able to output ProPruner modules directly, so you won't need to use the ProPacker and the modules won't need to be altered by the replay before playing.

BASIC USE OF THE PROPRUNER PLAYROUTINE

The first thing to do when using the ProPruner CIA shell or Level 6 timing is to fetch the vector base register and store in PPR_MainData+PPRM_VBR. The replay will then use it for compatibility on higher processors.

In order to use the ProPruner playroutine, you have to initially call PPR_Init with A0 pointing to your module and A1 pointing to your samplefile if the use of a samplefile is enabled. When this is done, you call PPR_Music each vertical blank to achieve music. CIA timing is described separately later on. When you want to stop the music, call PPR_End. It is perfectly safe to call PPR_Init and PPR_End while your VBlank interrupt is still enabled.

The function PPR_Init may also be used for restarting the module and all function trashes all registers except where else is stated. If noteplayers are disabled, the module must be in chip memory. If you use a separate samplefile, only the samplefile need to reside in chip memory.

PPR_Init should not be called while a module is playing without first calling PPR_End.

LEVEL 6 MODE

You may have ProPruner use the level 6 (CIA timers) interrupt for DMA waiting in order to save cycles when playing modules. Keep in mind, though, that neither the timers nor the interrupt vector is allocated from the system, so this feature should only be used when all hardware resources are available. When level 6 mode is in use, you cannot figure out the rasterline usage by timing the PPR_Music call since some of the actual work is done in an interrupt which occurs a while after the PPR_Music call. Level 6 timing is enabled by setting PPR_Level6 to 1.

SEPARATE SAMPLEFILES

ProPruner supports separate samplefiles in order to save precious chip-memory and delta-processing. Delta-processing is currently only available when separate samplefiles are enabled. The samplefiles are extracted from modules using the
PPRStrip and the PPRSampler commands.
To enable the usage of separate samplefiles, set PPR_SampleFile to 1.

PREDOUBLING

This feature is hard to explain, but shortly it means that the module is preprocessed during init in order to achieve slightly faster replay. Since this processing may cause errors in some VERY large modules, it is possible to disable it. Predoubling is on as default and may be turned off by setting PPR_PreDouble to 0.

MASTERVOLUME CONTROLS

ProPruner's mastervolume system is a combination of balance and mastervolume. The mastervolume provides a small speed overhead and should therefore be left off if unused. If mastervolume is enabled, call PPR_SetMasterVol with mastervolume for left side in D0 and mastervolume for right side in D1. The range is 0-64. To enable mastervolume, set PPR_MasterVol to 1.

DELTA-PROCESSED SAMPLEFILES

Delta-processing is currently only supported for separate samplefiles. Delta-processing is an altering of the sample data in order to make them easier to compress. When delta-processing is enabled, the ProPruner

playroutine will decode the sampledata back to their original form before starting the module. Delta-processing causes no changes to the final sound. With delta-processing enabled, the ProPruner replay will still handle module without delta-processing. To enable delta-processing, set PPR_Delta to 1.

SELECTABLE FINETUNE

The finetune option consumes both rastertime and space in the replay, so it is therefore advised to disable finetune if the replay is used to replay module(s) without finetune. The packer will inform you of wether the module uses finetune or not. To disable finetune, set PPR_Finetune to 0.

CIA TIMING USING THE PROPRUNER PLAYROUTINE

The ProPruner playroutine does not support CIA timing directly, but relies on the caller to provide the interrupt. If CIA timing is enabled, the user must himself allocate necessary resources and set up an CIA interrupt that calls PPR_Music. The playroutine will then call PPR_SetBPM with beats per minute in D0 each time the CIA tempo is supposed to change. The PPR_SetBPM function must be supplied by the caller and may only destroy D0 and D1. Please do also note that the Level6 function seldom works correctly when CIA mode is turned on. To enable the CIA mode, set PPR_CIA to 1.

SELECTABLE 68020+ OPTIMIZATIONS

The ProPruner can be assembled using special 68020+ optimizations, which may result in a slight rastertime usage decrease. The optimizations is mostly noticeable when powerpattern is disabled, but is always (to some extent) usable. To enable 68020+ optimizations, set PPR_68020 to 1.

PAUSE/CONTINUE FUNCTIONS

The ProPruner playroutine comes with selectable pause/continue functions. When these are enabled, PPR_Pause is used to pause the module and PPR_Continue is used for continuing the replay.

REWIND AND FAST FORWARD FUNCTION

Another feature of the ProPruner playroutine is the possibility to skip one position forward/backward in the module. When this possibility is enabled, PPR_Forward will skip one position forward in the module and PPR_Rewind skips one position backwards. If the module is at the beginning, ProPruner will automatically wrap to the end of the module. These function may cause overstep events and may miss position jumps!

THE POWERPATTERN OPTIMIZATION

The heart of ProPruner is the powerpattern feature. This is a way of

storing the patterndata that differs slightly from both ProPacker (which ProPruner is based on) and ProTracker. PowerPattern should pose any problems but may be disabled by setting PPR_PowerPatt to 0.

NOTEPLAYERS - WHAT?

If the noteplayer feature is enabled, ProPruner itself won't bang straight on the hardware to play the sound, but will use supplied functions and macros for achieving sound. The macros are:

PPRN_InitReg (Channel) - Is called to enable the noteplayer to set up A5 pointing to a data area needed by the other macros to perform correctly. The argument "Channel" is in the range 0-3, thus making the selected channel the "current" channel.

PPRN_KillSound - Has the sole purpose of killing the sound of all channels. This macro may be called without having A5 pointing to anything of significance.

PPRN_KillCh - Kill all sound on the current channel.

PPRN_TrigWave (Wavestart, Wavelen) - Causes the noteplayer to immediately start playing the selected wave and loops back to the start of the wave when the wave is through playing. Wavelen is given in number of words.

PPRN_SetWave (Wavestart, Wavelen) - Does nearly the same as PPRN_TrigWave except that the wave is not started until the current wave has reached it's end.

PPRN_SetVolume (Volume) - Sets the volume (range 0-64) for the current channel.

PPRN_SetRate (Rate) - Sets the samplerate for the current channel.

PPRN_SetMVolL (Volume) - Sets mastervolume for the left side. Range is 0-64. A5 is not initialized.

PPRN_SetMVolR (Volume) - Sets mastervolume for the right side. Range is 0-64. A5 is not initialized.

PPRN_Filter (State) - Attempts to change the status of the lo-pass filter. The noteplayer doesn't need to support this action.

At the beginning if PPR_Music, the function PPRN_MakeSound is called. This function may be used for mixing and trigging the actual sound output. To enable noteplayer, set PPR_NotePlayer to 1.

The noteplayer feature has not been debugged at all, so don't be surprised if it doesn't work. I would, however, be glad to hear about it anyway.

SELECTABLE NUMBERS OF CHANNELS TO PLAY

The ProPruner playroutine is able to play less than 4 channels in order

to set the remaining channels free for soundeffects. This is especially useful in conjunction with the SoundFX soundeffect replay engine. Disabling channels effectively saves a huge amount of raster time and should therefore be used even when the last channels isn't in use when the module to replay has no notes on the last channels. To use this feature, simply set PPR_Channels to the number of channels to play.

OBTAINING INFO FROM THE MODULE

The ProPruner playroutine supports a effectcommand not supported by other replay, effectcommand 8 to send events to the replay. When the module issues for instance effect command 847, the byte PPR_MainData+PPRM_Msg will contain the hexadecimal number 47. The programmer is free to read this value at any time and take actions upon it.

OVERSTEP EVENTS

If the module has reached the end, the byte PPR_MainData+PPRM_RestFlag will be set to true (non-zero). The caller may then take proper actions, such as loading another module.

1.97 Using SoundFX

WHAT IS SOUNDFX?

SoundFX is a sound-effect engine that enables playing of 2 samples through one normal audio channel. The mixing is fairly fast and provides minimal overhead. Both samples must have the same samplerate and cannot be looped in current version. This is done to save CPU cycles when mixing.

GENERAL USAGE

Call SFX_Init with the samplerate in D0. Minimum values are 123 for PAL and 124 for NTSC. When using the fastmode (described below) only rates of 226/228 or higher are supported.

Call SFX_SetVolume with volume (0-64) in D0 for adjusting the mastervolume. This is not set to default values and should therefore be done after the SFX_Init call.

Call SFX_StartSound with D0 stating which channel-partition to play on (0 or 1 for now), A0 pointing to the sampledata (should be in fastmem if possible) and A1 pointing to the end if the sampledata (samplepointer + samplesize). The sampledata should be halved to avoid distortion. In assembly this may be done by issuing a LSR.B #1 on all bytes.

Call SFX_End when the engine is not in use anymore. After this call, SFX_Init must be called before further sound effects may be generated.

While the sampledata does not need to reside in chip, the SoundFX engine

itself has to because of some buffers.

SETTING THE BUFFERSIZE

The buffersize tells SoundFX how many bytes to mix at a time. Changes to the sound (such as new samples) are only senced at the end of a buffer. Smaller buffers yields better response-times but provides more CPU overhead when starting the mixing while higher values does the opposite. The default setting of 1024 bytes should be sufficient. To change the buffersize set SFX__BuffSize to the number of bytes you wish the buffersize to be. Please note that the buffersize must be an even number.

THE FASTMODE

The fastmode provides a somewhat faster mixingscheme, but has worse audio output than the normal mode. When fastmode is active you can only use half the frequence of normal notes (rate 226/228 or higher) as normal. This mixing mode may produce high-frequency tones which may both distort the sound and destroy loudspeakers! To get rid of this quirk, enable the audio filter. In fastmode, however, the sampledata does not need to be halved. To enable fastmode, set SFX__FastMode to 1.

DIFFERENT OUTPUT MODES

The SoundFX engine at current supports two different output modes. To select one, set SFX__Mode to the number of your mode.

(0) - Vertical blank. This mode is perhaps the most convenient one for games that destroys the OS and does not wish to set up audio interrupt themselves. The engine relies on the user to call SFX_VBLHandle each vertical blank. All registers will be trashed. In this mode, the user must choose both buffersize and the replay rate so that one buffer takes exactly on frame to play. For help on how to calculate this, refer to the Hardware Reference Manual from Addison Wesley. For the rate 224, the samplebuffer should be set to 636 bytes.

(1) - OS Audio interrupts. (Default) This mode uses the operating system to set up an audio interrupt which handles everything. The sound replay is transparent to the caller and the buffersize may be freely chosen.

1.98 Using PTCalcTime

WHAT IS PTCALCTIME?

PTCalcTime is a source that calculates the playtime of ProTracker modules. This source operates internally with 1/25000ths of a second and should therefore be very accurate.

GENERAL USAGE

Call the function PTCalcTime with the registers set up as follows:

D0 - The songpos to start the module from. Normally, this should be 0.

D1 - 25000th of a second to pass between each interrupt. This should normally be set to 25000/50.

D2 - Set to 1 if the timing is CIA based. Otherwise, set to 0.

A0 - Points to the ProTracker module.

A1 - Points to four longwords that are used for storing the playtime. The first longword will contain hours, the next minutes, then seconds and finally 1/100 seconds.

The function returns shortly, but might use up to 2-3 seconds on really slow Amigas.

1.99 The TrackerTool command

What is TrackerTool?

TrackerTool is a small DOS-command designed for performing certain tasks on modules, which is clearly beyond the scope of ProTracker itself. This tasks mainly involve repairing and converting modules. (Current output modes are only normal ProTracker modules, though.)

Why use TrackerTool?

If you've tried QuadraComposer, you may have found out that QC is not able to identify Noise/Pro-Tracker modules properly if their size is wrong. To make this problem even worse, some tools save 4 bytes extra on modules because of a bug in earlier replay-routines. In order to fix these modules you either load them one by one into ProTracker (if they're not too damaged for ProTracker to recognize them!) and save them back to disk or you could use TrackerTool.

How to use TrackerTool?

The template for TrackerTool is:

TrackerTool SOURCE=FROM/A,ALL/S,VERBOSE/S,UPDATE=UD/S,WRITE=TO

SOURCE may be a file (ex: "mod.jugux"), a directory (ex: "modules/") or a wildcard (ex: "mod.eat#?").

ALL specifies that subdirectories is entered recursively.

VERBOSE will cause TrackerTool to print a lot of useless information about what it actually does. If TrackerTool shows strange behaviour, enable VERBOSE in order to find out what actually goes wrong.

UPDATE specifies that the original module(s) are to be overwritten with

the repaired module(s).

WRITE specifies a path to write the repaired module(s) to.

For repairing all modules on you device XST-00: you would write:

TrackerTool XST-00: ALL UPDATE

Or repairing all modules in the current directory starting with an "a"
and write the repaired modules to RAM: you would write:

TrackerTool mod.a#? WRITE RAM:

Anything more to tell me?

Since the main reason for using TrackerTool (at current) is repairing
possibly damaged modules, TrackerTool does no attempt to identify the
specified file(s). Because of this, you must NEVER run TrackerTool on
any files that are not 31-instrument tracker modules. This includes text
files, IFF pictures and 15 INSTRUMENT TRACKER MODULES! If you attempt
this, your Amiga might crash with a horrible wail and if used in
conjunction with the UPDATE keyword, you will probably lose your files.

TrackerTool does, however, support up to 256 patterns, DYN samples and
128 Kb long samples without problems.

1.100 ProPacker v2.1

READ FIRST!

ProPacker is needed to generate the modules for the ProPruner
playroutine. The ProPacker v2.1 is written by Christian Estrup and is
spread without his permission. What follows here are the original docs,
only slightly adjusted to fit the guidelines for this AmigaGuide
document. Some parts of the doc concerning the replay has been cut out.
Do also note that the ProPacker does not support being started from the
Workbench, but must be started from CLI/Shell.

PREFACE

Pro-Packer 2.1 is Freely Distributable. This means, that the program may
be freely distributed (!), as long as the replayer and this docfile are
included. NO money WHATSOEVER may be made from selling Pro-Packer 2.1.
Public Domain companies and their likes, who want to sell this program,
have to get a special, written permission from me. You will find my
address at the bottom of this doc-file.

INTRODUCTION

Pro-Packer 2.1 enables you to pack your ProTracker-modules (and
compatibles). It comes with a replayer, which is able to play the packed
modules a lot faster than the original PT-replay-routine.

THE PACKER

Using the packer is fairly easy. To load a module, just click in the load gadget (surprise). You will then see the familiar (?) req.library file-requester. (This means, of course, that you need req.library to run the packer!)

When you've selected a module, it will automatically be packed. When packing is finished, the packer will show you the length of the packed module. The packer will also tell you, whether or not the module uses finetune (see 'The Replayer' about this).

To save the packed module, just click in the save gadget (easy, isn't it?), and you'll see the file-requester again. Select a filename, and you're done...

COPYRIGHTS

req.library is (C) 1989 by Bruce Dawson and Colin Fox.

1.101 PPRStrip & PPRSampler

WHAT ARE THESE?

PPRStrip and PPRSampler are commands used for handling the generation of separate samplefiles for use with the ProPruner replay routine. These commands was originally thrown together for personal use, but should nevertheless be bugfree as they have been thoroughly tested. Both these commands needs OS 3.0+ to work.

PPRSTRIP

PPRStrip is used to strip the sampledata from ProPacker v2.1 modules. The only argument supported is the filename, so the original module will be overwritten.

PPRSAMPLER

PPRSampler is used for generating a samplefile from a ProTracker module. The command supports three arguments:

SOURCEFILE - The complete path and filename of a ProTracker module. This file will only be read and not changed in any way.

DESTINATIONFILE - The filename of the samplefile to generate.

DELTA - This is a keyword which, if present, causes PPRSampler to delta-process the sampledata. Look in the documentation of the

ProPruner
playroutine for information on delta-processing.

1.102 CIA tempo calculation

```
CIA TEMPO ISSUES

    Fcolor                          = 4.43361825 MHz
                                      (PAL color carrier frequency)
    CPU Clock   = Fcolor * 1.6      = 7.0937892  MHz
    CIA Clock   = Cpu Clock / 10    = 709.37892  kHz
    50 Hz Timer = CIA Clock / 50    = 14187.5784
    Tempo num.  = 50 Hz Timer*125 = 1773447

For NTSC: CPU Clock = 7.1590905 MHz --> Tempo num. = 1789773

To calculate tempo we use the formula: TimerValue = 1773447 / Tempo The
timer is only a word, so the available tempo range is 28-255 (++). Tempo
125 will give a normal 50 Hz timer (VBlank).

A normal ProTracker VBlank song tempo can be calculated as follows: We
want to know the tempo in BPM (Beats Per Minute), or rather quarter-
notes per minute. Four notes makes up a quarternote. First find
interrupts per minute:

        60 seconds * 50 per second = 3000

    Divide by interrupts per quarter note = 4 notes * speed
    This gives: Tempo = 3000/(4*speed)
    simplified: Tempo = 750/speed

For a normal song in speed 6 this formula gives: 750/6 = 125 BPM
```

1.103 DYN Samples

```
DYN SAMPLES

Normal 8 bit samples are in most cases not "real" 8 bit samples, since
the waveform very rapidly moves towards the zero line, i.e., amplitude
zero. It then oscillates between values of, let's say, -5 to +5 or even
worse, between -2 and +2, very often even between 0 and 1. Only the very
first part of the sample, the attack, is in the nominal 8 bit range, the
rest is more or less in 1 to 4 bit resolution only! Considering the
dependance of "1 bit = 6 db", the amiga has a signal/noise ratio of 48
db. This is not very good, even low budget tape recorders have a better
S/N ratio. Of course, only 16 bit systems like CD players or DAT
recorders really reach 96 db (16 bit -> 16*6 => 96), but the amiga is
clearly at the botton of the list. So if we have an effective 1 to 3 bit
resolution, the S/N ratio is as bad as 6 to 20 db :-(

If you ever wondered why your sampler produces noise even when sampling
directly from a CD, it is not your samplers fault. It is the 8 bit
```

resolution, which is not really an 8bit resolution! Same is true if you take very high quality 16 bit samples (yes, some privileged or stupidly rich people have access to 16bit samples :-) and convert them to 8 bit by using the higher 8 bit. The sound quality is very surprisingly not much better (is it better?) than with a good 8bit sampler.

This fact is known, everybody tries to compensate it. Some people do it by sampling far above 0 db level, which leads to a cutoff of the peaks in the attack phase of the sample. Some kind of sounds allow this rather rude technique, especially percussive sounds like bassdrum or snare, also basses sometimes. Many people do this also on other sounds (Hi Jim! ;-) to get "loud" samples. Clearly, this improves the sound quality at the end of the samples. It is more or less a matter of trial and error to get good samples this way, but as a workaround, I think this is a valid method.

Ok, so far concerning the situation of today's amiga sampling. But now to the DYN system (finally :)

HOW DYN WORKS

It utilizes 16 bit samples which are converted to special 8 bit samples. The 16 bit sample is divided into "parts", which are normalized to 0 db. 0 db means optimal recording condition, i.e. full usage of the 8 bit range. The amplificiation factors of each part are stored. On replay, the amiga hardware volume registers are set to compensate the amplification of each part, which then reproduces the original signal. The result is, that the sample data are always kept in the full 8 bit range, which leads to significantly less noise, especially on fading samples, samples with reverb, or any sample with a considerable dynamic volume change. Since the effective resultion changes dynamically between 8 and 14(!) bit, I called the system DYN, for sake of a short, describing name.

TECHNICAL DETAILS

The volume has to be altered while the sample is playing. It must be changed very accuratly between the different parts. The only way to realize this is by using the audio-interrupts. This was quite tricky, since there is no (good) documentation available concerning the programming of the audio-interrupts. However, to certain extent it works now, so I am happy. :-)

The DYN samples themselves are stored as normal IFF 8bit samples, plus an additional DYN chunk holding the information about the volumes each part must be played with, and the length of each part. The structure of a DYN chunk is described somewhere in the source. :-)

RESTRICTIONS OF THE DYN SYSTEM

Creation of DYN samples - First, it is difficult to create dyn samples.

One needs 16 bit samples to start with, not so easy for everyone of us. The only one existing conversion program, called "DoDyn", is an ugly CLI program which eats a lot of parameters, but has no GUI yet :-(One needs several tries to get part lengths which do not produce clicking when the part volumes are changed. A more intelligent DoDyn is needed!

Volume restrictions - Imagine a sample with a volume sequence like 64-10-5-3-1-1-1 (very typical!). If you play this sample with volume 32, you get either 32-5-2-1-0-0-0 (very bad) or you get 32-5-2-1-1-1-1 (better, but still not very good.). The volume ratios are distorted now, the amplitude envelope will not be the one of the original 16 bit sample. This is a result of the intrinsic volume changes, no way to overcome that. It is a question of DoDyn parameters, for example, not to allow amplifications which lead to volumes less than 4 when played with volume 64. this would look like 64-10-5-4-4-4-4 then, played with volume 32 giving 32-5-2-2-2-2 which is not so bad, but the max. resolution imporovement would be to 12 bit only. Thats why I called the DYN system also the "12 bit system".

Summarizing, playing dyn samples at low volumes can lead to distortion, because the original different volumes will get more or less the same.

Handling - Dyn samples cannot be modified easily regarding length. It is possible, but it needs enourmous efforts to keep track when cutting or inserting to the sample, since the DYN part table must be adapted. There is a limit for the length of parts! Each part must be long enough to last over the DMA wait, for example. The DMA needs some time to copy the data to the internal registers, so the interrupts must not come to frequently. Therefore, handling of DYN samples is not very comfortable.

Timing - When a sample is played on 16 khz (around C-3), there are only 320 sample points played in a 50 Hz (20 ms) frame, which is not much. It means, that every scan line, another point is played. The hardware always eats 2 points at once, so every 2 rasterlines, there is a change. The Dyn interrupt itself lasts ca. one 4th of a rasterline. There can be 4 interrupts which want to act at the same time, and it is clear that they are not allowed to interrupt each other, so they have to wait until the one before is ready. Since the soundroutine, running with CIA timer, may not interrupt the audio-interrupts (which have a lower priority, btw), the timing will be not *that* exact, but the largest delay between each music routine frame will be about 1 rasterline only, therefore negliable.

For those who think, the interrupts could probably miss the part changes, don't worry about that. The interrupts DO NOT come after a part is played, but after the data are copied into the internal registers by the audio hardware. The data of the next part are played by the hardware automatically, so the correct sequence of the parts is granted. BUT, and now comes the BUT, the VOLUME is set by the interrupts at the time the interrupts come! What the interrupts do, is setting the volume into the audio volume register, which is NOT buffered, but a "direct" register! Because of the buffering of the DMA registers, the interrupts set the volume of the previous part which was set into the hardware concerning part start and part length. This is the crucial point. Now, an interrupt really can come too late concerning the volume. It can be too late for the time of 3 interrupts, because the first thing the interrupt does, is setting the volume of the previous part, which is now played (assuming

the special case, all four interrupts want to come at the same time). If the part volumes do big jumps, it can happen that some points of the new part are still played with the old volume, which is definitely too loud. There is no solution for this problem, but there is a workaround. It is up to DoDyn not to allow big volume jumps, so the annoying clicking is not that bad.

However, the chance to have interrupts waiting is almost negligable. Normally, the partlengths of different samples differ quite much, and you play them at different frequencies, so the interrupts should never have to wait. You can force this problem to appear by playing the same dyn sample at all 4 channels at the same frequency.

1.104 Index

Index of database Default

Documents

Adjust tempo effect command

Arpeggio effect command

Chronicles and anecdotes

CIA tempo calculation

Contacting the authors

Cut note miscellaneous effect command

Delay note miscellaneous effect command

Delay pattern miscellaneous effect command

Development and progress

Disk operations

DYN Samples

Effect commands

Fine volumslide down miscellaneous effect command

Fine volumslide up miscellaneous effect command

Fineslide down miscellaneous effect command

Fineslide up miscellaneous effect command

For the technical minded

Frequently asked questions

Gadgets

Glissando control miscellaneous effect command

How to create music using PT

How to start ProTracker

Introduction

Invert loop miscellaneous effect command

Keyboard

Keyboard percussion

Known bugs

Legal mush

Main screen - Chord editor

Main screen - Edit gadgets

Main screen - Edit options

Main screen - Filter editor

Main screen - Sample edit

Main screen - Sample section

Main screen - Song section

Main screen - Various

Miscellaneous

Nice to know

Other tracker-clones

Pattern break effect command

Pattern loop miscellaneous effect command

Patterns

Pointer colors

Position jump effect command

Position table

PPRStrip & PPRSampler

ProPacker v2.1

ProTracker documentation

ProTracker IFF song&modules

ProTracker song&module format

Requesters and I/O

Retrig note miscellaneous effect command

Sample offset effect command

Sampler - Main

Sampler - Volume requester

Samples

Set filter miscellaneous effect command

Set finetune miscellaneous effect command

Set volume effect command

Setup screen - Flags

Setup screen - Miscellaneous

Slide down effect command

Slide up effect command

Standard paths

Thanks and acknowledgements

The TrackerTool command

Things we will implement

Things we'll never implement

Toneportamento and volumeslide effect command

Toneportamento effect command

Tremolo control miscellaneous effect command

Tremolo effect command

Using PTCalcTime

Using SoundFX

Using the PPR playroutine

Using the PT playroutine

V0.89 BETA

V1.0A

V1.0B

V1.0C

V1.1A

V1.1B

V1.2

V1.2C

V1.3B

V1.8

V2.0

V2.1A

V2.2A

V2.3A

V3.00 BETA

V3.01

V3.10 BETA

V3.14B

V3.15

Version history

Vibrato and volumeslide effect command

Vibrato control miscellaneous effect command

Vibrato effect command

Volume slide effect command

What is a module?

What is DYN samples?

What is ProTracker?
Buttons

3~+~A

4~+~A

accurate~timing

authors

Break~pattern

C-Command

cheating

Chord~editor

Chronicles~and~anecdotes

CIA~tempo~calculation

command~1

contact~the~authors

Contacting~the~authors

Cut~note

Delay~note

Delay~pattern

Disk~operation~screen

Disk~operations

disk~operations~screen

disk~operations,

disk-operations

disk-operations~screen

DYN~sample

DYN~samples

DYN~sound

DYN~technical~chapter

DYN-sample

DYN-samples

DYN-Samples

E4x

E7-

Edit~gadgets

Edit~options

edit-gadgets

Effect~commands

effect-command

Effect-command~9

F~command

Filter~editor

Fineslide~pitch~down

Fineslide~pitch~up

Fineslide~volume~down

Fineslide~volume~up

Flags

For~the~technical~minded

Frequently~asked~questions

Frequently~Asked~Questions~section

Gadgets

Glissando~control

How~to~create~music~using~ProTracker

How~to~start~ProTracker

inform~us

Introduction

Invert~loop

Jump~to~position

Keyboard~percussion

Keyboard~shortcuts

Known~bugs

launch~PT

Legal~mush

Main

main~screen

Miscellaneous

Miscellaneous

Nice~to~know

None/arpeggio

normal~portamento~down

normal~portamento~up

normal~volumeslide

Pattern~loop

patternbreak~effect~command

Patterns

Pointer~colors

Position-table

PPRStrip~and~PPRSampler

PPRStrip and the PPRSampler

ProPacker~v2.1

ProPacker~v2.1~by~Christian~Estrup

ProPruner

ProTracker~development~and~progress

ProTracker~IFF~song/module~format

ProTracker~song/module~format

ProTracker~version~history

ProTracker-relevant~clones

replen

Requesters~and~I/O

Retrig~note

Sample~edit

Sample~edit~operations

sample~editor

Sample~offset

Sample~section

sampler

Sampler~screen

Samples

select~instrument

send~them

separate~chapter

Set~finetune

Set~low-pass~filter

Set~speed

Set~volume

Setup~Screen

setup-screen.

Slide~pitch~down

Slide~pitch~up

slide~the~volume

Slide~volume

sliding~the~volume

Song~section

speed~of~the~song

Standard~paths

starting~PT

the~three~gadgets

The~TrackerTool~command

Things~we~will~implement

Things~we'll~never~implement

tone-portamento

Toneportamento

TrackerTool

Tremolo

Tremolo~control

tunetone~setting

Using~the~ProPruner~play-routine

Using~the~ProTracker~play-routine

Using~the~PTCalcTime~playtime~calculator

Using~the~SoundFX~sound-replay~engine

V0.89~BETA

V1.0A

V1.0B

V1.0C

V1.1A

V1.1B

v1.2

V1.2C

V1.2D

V1.2E

V1.2F

V1.3A

v1.3A~and~v1.3B

V1.3B

V1.8

v2.0

v2.1A

V2.2A

v2.3a

v3.0

V3.00~BETA

V3.01

v3.10

V3.10~BETA

V3.14B

V3.15

Various

version~history

Vibrato

Vibrato~control

Volume~requester

What~is~a~module?

What~is~DYN-samples

What~is~ProTracker?