The HUGETracker  
manual

introduction

Hi, this is the manual to hUGETracker. I wrote this program because there wasn’t a music editing tool for the Gameboy which fulfilled the following requirements:

* Produces small output
* Tracker interface
* Usable for homebrew titles
* Open source

But now there is!

I’d like to acknowledge

1. Christian Hackbart for creating UGE, which serves as hUGETracker’s emulation core
2. Rusty Wagner for writing the sound code which was adapted for UGE
3. Lior “LIJI32” Halphion for SameBoy, a super-accurate emulator which I used for debugging and copied the LFSR code from
4. Declan “Dooskington” Hopkins for GameLad, which I yanked the timing code from
5. Eldred “ISSOtm” Habert, who helped me navigate the Gameboy’s peculiarities and for writing an alternative sound driver for the tracker
6. Evelyn “Eevee” Woods, whose article on the Gameboy sound system was valuable in writing the music driver.
7. B00daW, for invaluable testing and debugging support on Linux.
8. The folks who created RGBDS, the assembler used for building ROMs from songs.

I hope you enjoy composing in hUGETracker, and if you make any cool songs, I’d love to hear from you and potentially include them as demo tunes that come with the tracker.

E-mail me at [yux50000@hotmail.com](mailto:yux50000@hotmail.com) and get in touch!

-Nick “[SuperDisk](http://nickfa.ro)” Faro

Contents

[Prerelease information 4](#_Toc30966709)

[Concepts 5](#_Toc30966710)

[Interface 6](#_Toc30966711)

[VU Meters and Oscilloscopes 7](#_Toc30966712)

[Tracker Grid 7](#_Toc30966713)

[Effect reference 8](#_Toc30966714)

[Hotkeys 10](#_Toc30966715)

[Routines 11](#_Toc30966716)

[Miscellaneous 12](#_Toc30966717)

[The clipboard 12](#_Toc30966718)

# Prerelease information

This is a prerelease version of hUGETracker. Most features are implemented, and most bugs are gone, but of course it’s not completely done. Notably, the master volume effect command, and routines are not currently implemented (but will be soon!). Also, hUGETracker doesn’t have a super-cool scene logo to show in the help -> about section, so if you have art skills and want to help out, get in contact ;)

There aren’t many good example songs yet as well, so send your songs in and they might be included with the next release!

Also, be sure to save often in case there’s a crash. There shouldn’t be anything that can crash the tracker, but better safe than sorry.

This manual isn’t complete, but the effect and hotkey reference are here, so that’s the most important part.

Have fun!

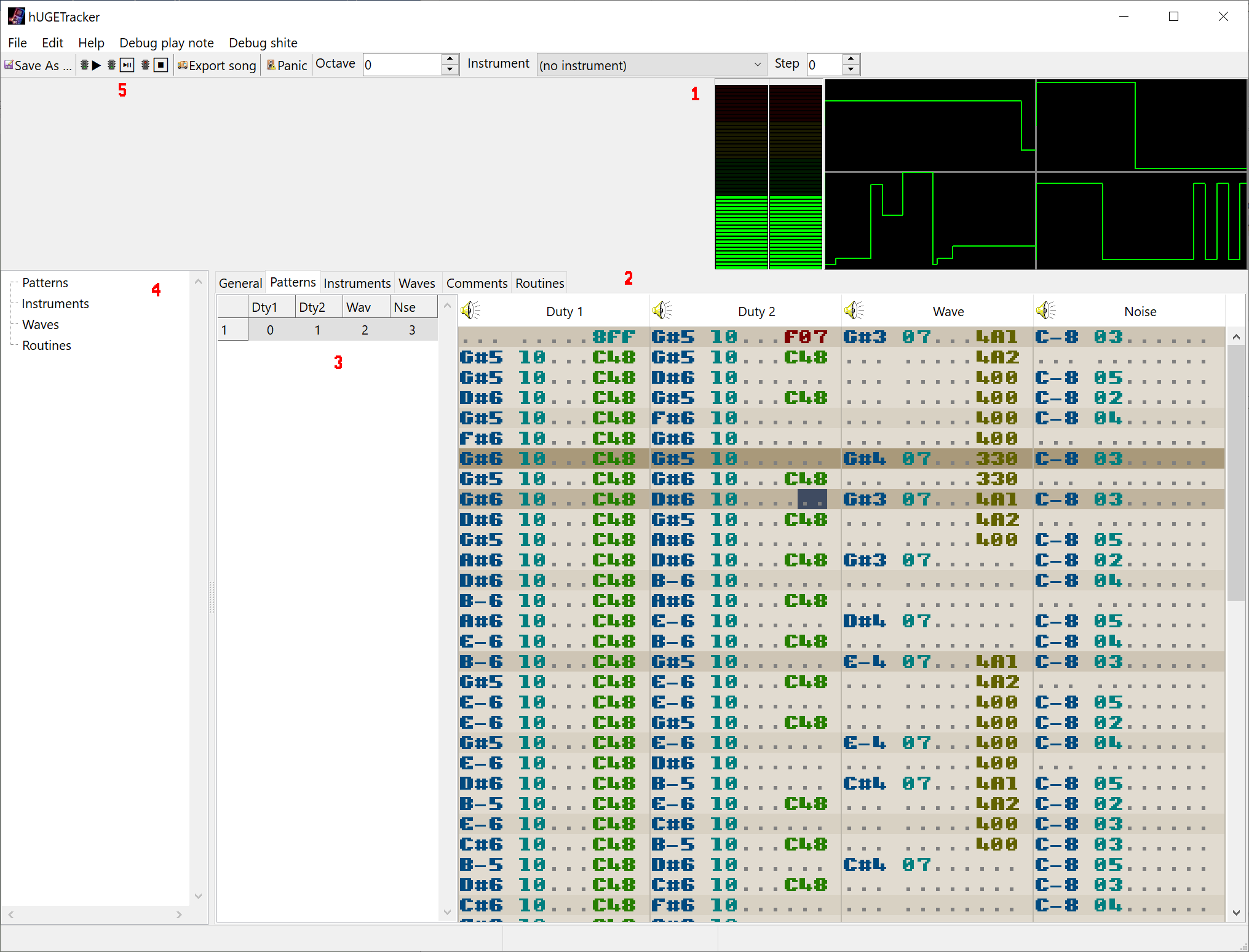
-Nick

# Concepts

To be written

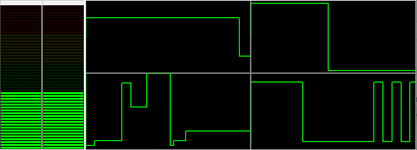
# Interface

The hUGETracker interface is styled similarly to conventional trackers such as ModPlug Tracker or ProTracker. If you’re comfortable composing in a tracker interface, then you’ll feel right at home.



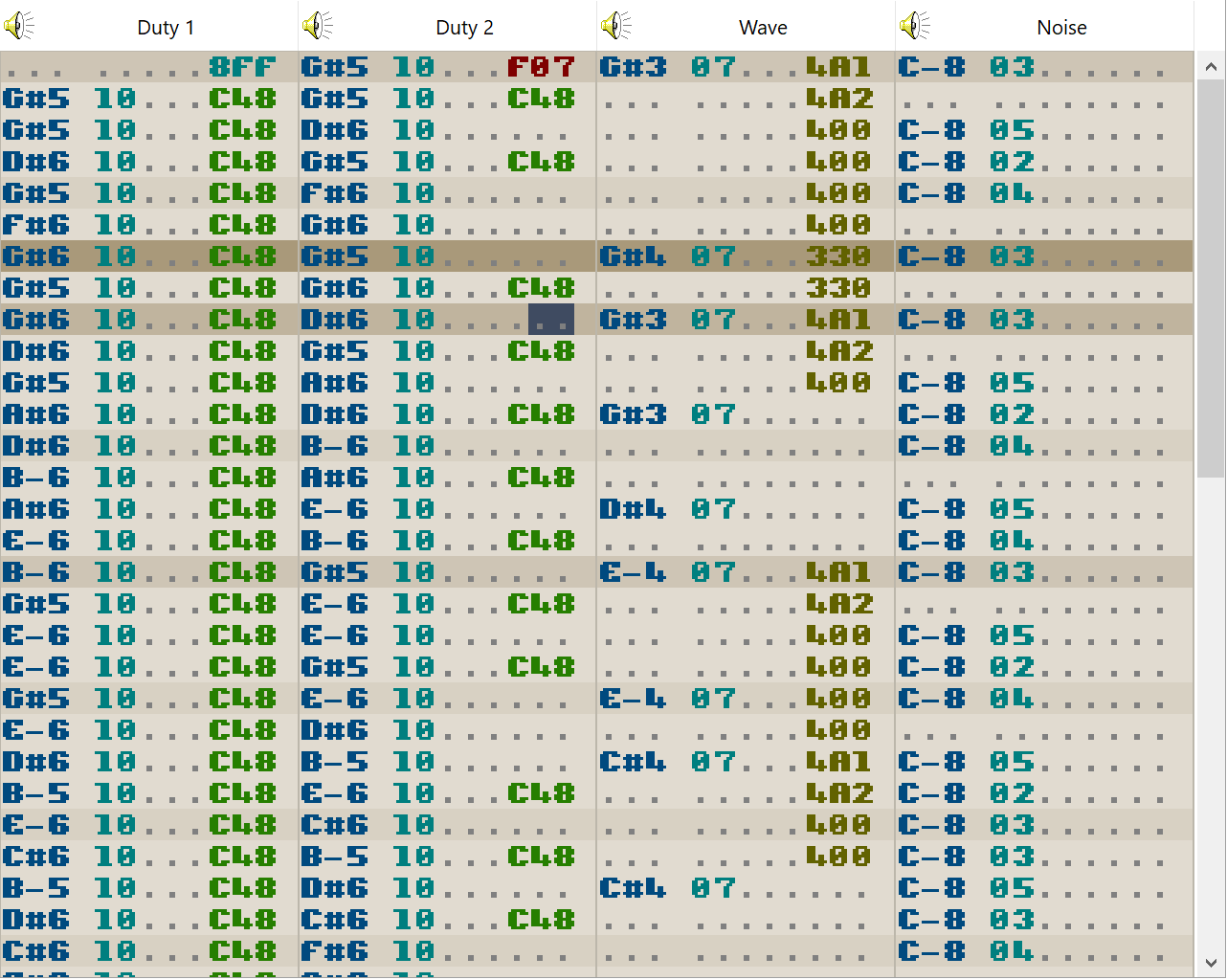
1. VU Meters and Oscilloscopes
2. Tracker Grid
3. Order Editor
4. Song components
5. Toolbar

## VU Meters and Oscilloscopes



The VU meters show the volume level for the left and right speakers. When volume gets too loud, they display as yellow/red. The oscilloscopes show the waveforms generated by the four Gameboy channels, duty 1, duty 2, wave, and noise.

## Tracker Grid



# Effect reference

The hUGETracker effect codes are intentionally similar to ProTracker or FastTracker’s. If you know them, then many of these effects will look familiar to you.

|  |  |  |
| --- | --- | --- |
| Effect | Name | Description |
| 0xy | Arpeggio | On every tick, switch between the playing note, note + x, and note + y, where `x` and `y` are values in semitones. Can be used to create “chords” or a strum effect. |
| 1xx | Portamento up | Slide the pitch up by `xx` units every tick. |
| 2xx | Portamento down | Slide the pitch down by `xx` units every tick. |
| 3xx | Tone Portamento | Slide the pitch towards the specified note value by `xx` units every tick. Stops when it reaches the specified note value. **This effect cannot be used in a cell with an instrument value**. |
| 4xy | Vibrato | Rapidly switch between the specified note value and note + y, at the rate of `x`, where `y` is a value in units. Valid values for `x` are 0, 1, 3, 7, and F. This is similar to arpeggio, except you can control the frequency, and the amount is specified in units rather than semitones. |
| 5xx | Set Master Volume | Sets the master volume control of the Gameboy for the left and right speakers. Use the effect editor to create one of these effects. Note that a volume of zero is not completely silent. Currently not implemented. |
| 6xx | Call Routine | Call a user-defined routine. See the section Routines. Will crash the song if an invalid routine is specified. |
| 7xx | Note Delay | Wait `xx` ticks before playing the note in this cell. |
| 8xx | Set Panning | Sets which channels play on which speakers. Use the effect editor to create one of these effects. Can also be used as a mute for a channel by setting it to output on neither left nor right. |
| 9xx | Set Duty Cycle | Select duty cycle for either channel 1 or channel 2. If this effect appears on the noise or wave channels, it will affect channel 2. Valid values for xx are 00, 40, 80, C0. Under the hood, the `xx` value is loaded directly into ch1 or ch2’s length register, so you could theoretically achieve other effects than just duty cycle changing. |
| Axy | Volume Slide | Slide the note’s volume up by `x` units, and then down by `y` units. This effect actually retriggers the note on each tick, which might not be noticeable for instruments without length/envelope, but could potentially sound bad if those are present. Recommended to use either instrument envelopes, or the `C` command instead if you can. **This effect does not work in the same cell as a note/instrument!** |
| Bxx | Position Jump | Jump to order `xx`. |
| Cxx | Set Volume | Set the volume of the channel to `xx`. **Must be accompanied by a note and instrument to work (except on channel 3).** Valid values range from 00-0F. |
| Dxx | Pattern Break | Jump to the next order, and start on row `xx`. |
| Exx | Note Cut | Cut the note short after `xx` ticks. |
| Fxx | Set Speed | Set the number of ticks per row to `xx`. Can be used in an alternating fashion to create a swing beat. |

# Hotkeys

The keyboard interface to hUGETracker is intentionally designed to be similar to ModPlug’s. If you’re familiar with it, then most of these keybindings will look familiar.

|  |  |  |
| --- | --- | --- |
| Hotkey | Action | Remarks |
| Ctrl-C | Copy | Copies the selected pattern data into the clipboard |
| Ctrl-X | Cut | Copies the selected pattern data into the clipboard, then erases the selected pattern data |
| Ctrl-V | Paste | Pastes any stored pattern data in the clipboard |
| Shift-V | Flood paste | Continually pastes stored pattern data one after the other until reaching the bottom of the pattern. Useful for repeating beats/phrases/swing tempos. |
| Ctrl-Q | Semitone up | Transposes the selected notes one semitone up |
| Ctrl-A | Semitown down | Transposes the selected notes one semitone down |
| Ctrl-Shift-Q | Octave up | Transposes the selected notes one octave up |
| Ctrl-Shift-A | Octave down | Transposes the selected notes one octave down |
| Ctrl-L | Select channel | Selects the entire pattern that the cursor is located in |
| Del | Erase | Erases the selected note data |
| Ctrl-Z | Undo | Undoes the previous action. |
| Ctrl-Y | Redo | Redoes the action last undone. |

# Routines

Routines allow you to implement your own effects. Currently routines aren’t implemented in the interface, but will be soon.

# Miscellaneous

Here are some random things about hUGETracker.

## The clipboard

The clipboard format for hUGETracker is compatible with ModPlug/OpenMPT! This means that you can convert your existing .mod, .xm, .s3m, or .it chiptunes into Gameboy tunes very quickly by just copying and pasting your patterns into hUGETracker. Note that the effects are not converted when pasted, so you’ll need to adjust the effects to work in hT.

