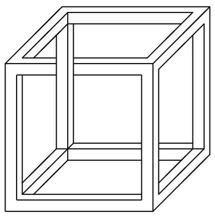
OPENCP 2.6pre6

Open Cubic Player 2.6pre6 Documentation

Felix Domke Fabian Giesen Tammo Hinrichs Dirk Jagdmann

September 1, 2006

Foreword



OpenCP is a music file player. It is derived from the *Cubic Player 2.0αppe* which was developed by Niklas Beisert. The latest version of is 2.6pre6. See chapter 6 on how to get a copy of OpenCP.

OpenCP is published under the GNU General Public License (GPL). That means you can use this software without charging for it. However OpenCP is not in the public domain! The source code is available to the public and everybody is encouraged to make changes to it. Please send any modifications and bugfixes to the authors. The authors do not take any warranty for any damages whatever kind produced by this software. For further details see the file `COPYING.TXT`

Contents

Part I.

Users Manual

Chapter 1

Installation and files

OPENCP needs no special install procedure. Simply copy all files you found inside the archive into one directory. If you like you can add this directory to your path statement. Example:

Install OPENCP into `C:\OPENCP` by typing:

1. `C:`
2. `CD \`
3. `MD OPENCP`
4. `CD OPENCP`
5. `PKUNZIP -D C:[theRightDirectory]OPENCP`

Then you can modify your `autoexec.bat` and add the new directory to the path statement. The new path might look like
`path c:\windows;c:\windows\command;c:\opencp`.

The following files are (at least) required to start OPENCP

cp.exe	the startup file used to start the OPENCP runtime system
cp.ini	the initializition file read by cp.exe to configure itself
cp.pak	the DLL library where all modules are stored

As OPENCP is a 32bit program it normally needs the DOS4GW extender to use protected mode. If the file `dos4gw.exe` could not be found in the current directory and the path, you get an error message.

When OPENCP loads a module the information stored inside the file is read and stored in a special file (the *module information cache*). Actually the cache splits up into three files:

cparcs.dat	Information about archives and files stored inside
cpmdztag.dat	Paths to `.MDZ`-files (see appendix 10)
cpmodnfo.dat	Informations about various modules

If you delete these files all file information gathered is permanently lost.

The `doc` directory contains documentation files in various formats and release notes etc. You can safely delete this directory if you are running out of disk space.

1.1. OPENCP and Windows 9x

OPENCP will run happily in a DOS-Box of Windows 95,98 and ME. You can use the native drivers of OPENCP, but this is not recommended with windows. If you have a sound card supported by windows and installed a

DirectX 5 or later you can use the windows drivers with OpenCP. This way OpenCP will work with every sound card supported by windows, which includes modern pci devices.

Installation is totally simple:

1. copy the `vxdapc.vxd` driver and the `vocp.dll` library into your texttt{windows“system di-rectory
2. run the `cphost.exe` program before starting OpenCP

`cphost.exe` will place an icon in your icon tray to indicate proper operation. If you use OpenCP regularly you can start `cphost.exe` with your autostart group.[1]

If a yellow exclamation sign is added to the icon (in your icon tray), something failed. Please contact us for a bug report together with system information, the log from `cphost.exe` etc.

If you have problems please see our section about the vapc driver in the FAQ.[2]

1.2. OpenCP and Windows XP

OpenCP will run under Windows XP if you use the VDMSound SoundBlaster Emulation. This program will emulate a SoundBlaster in the CMD Box of Windows XP which is routed to your native soundcard, which can be of any type supported by WinXP.

After you have installed VDMSound you have to start a command line box (e.g. *Start* → *Run* → *cmd*), change to the folder where VDMSound was installed and run *dosdrv*. Then the SB emulation is set up for the current CMD box. Then you have to change into the OpenCP folder and can start OpenCP normally.

To use the graphic modes you have to disable the VESA extensions in the `cp.ini` file. Please set the *uselfb* option to *no*. You can also use our optimized WinXP cp.ini.

Note that version 2.1.0β of VDMSound will crash if you try to use any of the graphic modes (oscillator, graphic spectrum analyzer, note dots etc.).

1.3. OpenCP and Linux

A Linux version of OpenCP is currently under development. Playback via the Open Sound System (OSS) linux sound drivers (which is course included the ALSA OSS emulation) is currently supported and textmode output to the linux console and ncurses. Graphic modes work via 8bit X dgm. Due to the use of the original assembly routines this software is currently for i386 only. Please use the current α development versions from `http://labs.nixia.no/ocp.php`.

1.4. OpenCP and other Operating Systems

You can run OpenCP with any operating system which supports the DOSBox Emulator. This currently includes most i386 unixes, many other architectures with a unix like OS (macintosh) and most versions of Windows.

To use the graphic modes you have to disable the VESA extensions in the `cp.ini` file. Please set the *uselfb* option to *no*. You can also use our optimized DOSBox cp.ini.

1.5. Notes for german users

Auf der deutschen Tastatur weichen die Tastenbezeichnungen bei den Sonderzeichen von denen der englischen Tastatur ab. In dieser Anleitung wurden dabei immer die englischen Bezeichnungen verwendet. Mit der folgenden Tabelle sollten aber alle Unklarheiten beseitigt sein.[3]

[1]Either move it into the autostart folder, or place a link in there.
[2]page 41
[3]Verwechseln Sie nicht die $\boxed{\longleftarrow}$ (Backspace) mit der $\boxed{\leftarrow}$-Taste! Die $\boxed{\longleftarrow}$-Taste liegt normalerweise über der $\boxed{\hookleftarrow}$-Taste (Enter/Return).

english	–	deutsch
SHIFT	–	⇑
CTRL	–	STRG
Ins	–	Einfg
Del	–	Entf
Home	–	Pos1
End	–	Ende
Pgup	–	Bild↑
Pgdown	–	Bild↓
Backspace	–	⟵
Enter	–	↩
Space	–	Leertaste
Tab	–	|←

Chapter 2

Starting OpenCP

After starting OPENCP from the command line the dos extender is loaded. By default OPENCP looks for a file `dos4gw.exe` in all directories included in the path statement and the home directory. When it cannot find this file, OPENCP exists with the error message `could not execute: No such file or directory`.

OPENCP then tries to detect the sound cards defined in the `cp.ini` file. If the card is not found you cannot use it. If more than one sound card is installed and found by OPENCP the first one listed in the `cp.ini` is used as the default device.[1]

By entering a filename as parameter onto the command line you can load a module. If a filename is given the file will be loaded and OPENCP starts with the player.[2] When the parameter is the name of a directory the fileselector starts with the given directory. If the parameter equals a certain archive all files found inside the archive will be played (by adding them to the playlist).

If more than one parameter is given a playlist will be set up, containing all found files. You can mix filenames of archives, files in archives and normal files on the hard disk.

You normally don't have to supply the right file extensions, as they are guessed by the fileselector. If the filename is not complete the file will not be processed (unlike the quickfind function).

OPENCP can be configured using the command line, although the configuration through the `cp.ini` file is more comfortable. The options split up into three main sections each starting with "-" followed by a letter. For each section different options are given, which will configure the player accordingly. You can precede each option with the section prefix every time or supply multiple options seperated with ",". Some special options do not require the use of a sections prefix, like the *help* switch.

The command line looks like:
`cp [prefix option[,option]] [specialOption] [filename]`[3]

Special options include:

`-h`	show a help screen
`-m`	use a monochrome graphics card (hercules or mda). Useful when starting OPENCP under windows.
`-c<name>`	use a configuration defined in `cp.ini`

Fileselector options are envoked with `-f`. The values in square brackets define a choice that must be made when using one of these options.

`r[0|1]`	remove played files from playlist
`o[0|1]`	don't scramble playlist order
`l[0|1]`	loop modules

Playback options are preceded by `-v`. Values in sharp brackets define a range in which the value must be taken.

`a<0...800>`	set amplification
`v<0...100>`	set volume

[1]You can change this in the player by using the `@:` device. See page 12 for detail.
[2]This will work if the file is inside an archivr located in the current directory. If no (unpacked) file is found all archives in the current directory are searched for an appropriate filename.
[3]Square brackets [] indicate an optional item that can be repeated.

`b<-100...100>`	set balance
`p<-100...100>`	set panning
`r<-100...100>`	set reverb
`c<-100...100>`	set chorus
`s[0|1]`	set surround
`f[0|1|2]`	set filter (0=off, 1=AOI, 2=FOI)

Device setting are accomplished with the suffix `-s`.

`p<name>`	use specific player device
`s<name>`	use specific sampler device
`w<name>`	use specific wavetable device
`r<0...64000>`	sample at specific rate
`8`	play/sample/mix as 8bit
`m`	play/sample/mix mono

Finally an example to illustrate the above features:

```
cp -fl0,r1 -va80,p50,f2 -spdevpdisk -sr44100 ftstar.xm
```

This will start OPENCP and load the file `ftstar.xm`[4]. The music will be played once and will not loop (`-fl0, r1`). Further the player is advised to amplify this file with 80%, set the panning to 50% and interpolate every sample (`-va80, p50, f2`). The mixed output will be saved into `.wav` format through the *diskwriter* device (`-spdevpdisk`) with a sample rate of 44.1KHz (`-sr44100`).

You can burn this WAV file directly onto a CD-Audio and play it with every normal CD player. A much simpler and more convinient way to make such a *sample image* of a module is by using predefined configurations with the `-c` switch. Have a look at the section *Using the diskwriter* on page 21.

[4]A marvelous piece of music composed by KB which won The Party 1997.

Chapter 3
Fileselector

If OPENCP is started without any command line arguments the fileselector will be loaded. With this powerful tool you can browse through your modules and set up playlists to be processed by the player. If you switch to the player the selected files will be loaded and processed.

Files can be stored inside compressed archives to save space on the hard disk. Those files are automatically unpacked to a temporary directory before scanning or loading. If you have many modules you should use this feature, as storing modules inside archives is totally transparent when using the fileselector.

3.1. Main screen

The fileselector splits into three main windows: directory list, playlist and module information as shown in figure 3.1.

The path window shows the current path and file mask. If you started OPENCP from the home directory you will get the following: `C:\OPENCP\*.*` means that the current directory is `OPENCP` on your `C` harddrive and all files are shown (`*.*`). You can edit the path and the mask by pressing (CTRL)+(Enter). After editing the path press (Enter) to change to the appropriate directory. You can edit the file mask to include only some files.

> `C:\*.mod` will change to the root directory of hard disk `C` and show all files ending with `.mod`. The default setting should be the current directory with a file mask `*.*` to show all files.

The most important window is the directory list. Here you can see all files in the current directory. If the extension is known to OPENCP the file information will be shown in different colors depending on the file type. Files not known to OPENCP will be shown in standard grey.

Leftmost the file name provided by the operating system is shown. The extension `.???` will specify the file type. The next column shows the title of the file if the file type includes a title. In the third column the number of channels is displayed. Finally rightmost the filesize is shown in bytes. If the module is included in a ZIP archive the *real* file size is displayed.

Right to the directory list you can see the play list. All files listed in this window will be played, after you change into the player. The order of entries in this window determine the order in which files are loaded unless you have enabled the *random* option.

The window at the bottom is the module information. Many music formats can store general information which is displayed here. If the file type does not support those information you can edit the fields inside this window manually and OPENCP will store the information for you.

Finally at the very bottom is the quick find feature, which lets you easily find files in the current directory.

3.2. Usage of the fileselector

The directory list shows you all files in the current directory which fit to the file mask set in the path window and OPENCP can detect. Under the alphabetically sorted files the directories and drives are shown.[1] Use the (↑) and (↓) to browse through the files. If you press (Enter) the selected file will be loaded and played with the player. Pressing (Enter) while selecting a

[1] `@:` is a special drive which lets you configure OPENCP without editing the `cp.ini` file, see page 12.

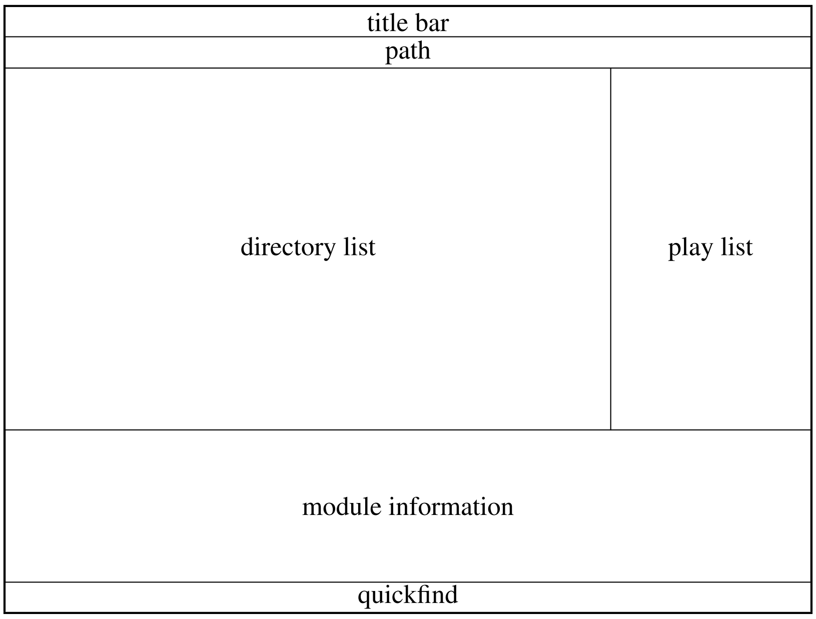


Figure 3.1.: the fileselector screen

directory or drive will switch to the selected item and the directory will be read. Pgup, Pgdown, Home and End will work as expected.

If a module is played and you are in the player f will beam you to the fileselector. You can always leave the fileselector by pressing Esc twice! If no module is playing the program will exit, while you will get back to the player if a module is played in the background.

Playlists are shown in the playlist window at the right side of the screen. The currently selected file is appended to the playlist by pressing → or Ins. ← or Del will remove it again. You can insert files multiple times into the playlist by pressing the appropriate keys more than once. If you have files in the playlist exit the fileselector by pressing Esc! This might seem confusing in the beginning, but you will notice the logic very soon. In the player you start the next song in the playlist by pressing Enter.

Normally you will start the fileselector from the player by pressing f. The current module will continue playing in the background. After you have selected a file you have to choices:

- Enter will stop the currently played module and load the selected one. Then you will get back to the player. Use this key if you want to play the selected module immiediatly.
- Esc will change to the player. Then you can start the next songs in the playlist by pressing Enter. If you have inserted files into the playlist use this key to exit the fileselector.

All files in the current directory will be inserted into the playlist by pressing CTRL+→ or CTRL+Ins. The playlist will be deleted by pressing CTRL+← or CTRL+Del.

Although in the playlist window only the filename is shown, OPENCP stores the complete path information. So you can insert files into the playlist from totally different directories and drives. If files are inserted into the playlist you can change to the playlist window by pressing Tab. Inside the playlist window all keys have full functionality. So you can load the selected module immiediatly by pressing Enter or remove the file from the list by pressing ←. If you are in the playlist window you can move the currently selected file by pressing CTRL+↑ and CTRL+↓. This will affect the order in which files are processed. CTRL+{Pgup,Pgdown,Home,End} work as expected.

If many files are inside a directory selecting a module with the cursor keys can be annoying, because it takes a long time to browse through the list. If you know the filename you can start typing it on the keyboard. This enables the quickfind feature. Characters already typed are shown in the quickfind window. The current directory is searched for files matching the typed characters. Often you don't have to type the complete filename, as it can be already determined by the leading chars. The typed characters must not fit the file exactly as small errors are neglected.

At the bottom of the screen the fileinformation window is located. If the file includes any additional information it will be shown at the appropriate fields. You can edit each entry manually. All module information is read by the fileselector once if it runs along this module the first time. The data is stored in three files located in your home directory of OPENCP refered as the *module information cache*. If the fileselector scans a directory and finds a module already stored in the module information cache it will use the information found in the cache. This way you can overwrite the information fields of the module. If you change to a directory which has not been processed by the fileselector it may take some time to read all file information out of the files and store them in the module information cache.

To switch to the module information window press SHIFT+Tab. You can use the cursor keys to select the entries. After pressing Enter the information can be edited. When pressing Enter again the changes are stored in the module information cache. Note: *Do not change the entry* type *as the file could not be loaded properly when the wrong filetype is entered! Normally you never have to change this entry, except for old 15 instruments amiga noisetracker modules!*

3.3. Advanced usage

The appearance and behaviour of the fileselector can be edited in the `cp.ini` (page 27) file or by pressing ALT+c. Changes made to the `cp.ini` are permanently, while configuration applied with ALT+c is only valid while OPENCP is running.

Afer pressing ALT+c you can toggle 13 options with keys 1...9 and a...d. The following list will explain every option:

1 *screen mode:* you can change the screen mode for the fileselector. 80x25 and 80x50 are standard screen modes and should be available on every vga card. 80x30, 80x60, 132x25, 132x30, 132x50 and 132x60 are only available with a proper VESA bios installed.

2 *scramble module list order:* if this options is enabled the files inside the playlist will be played in random order. Otherwise the order shown in the fileselector from top to bottom will be used.

3 *remove modules from playlist when played:* normally you will want this enabled as modules are only played once. If you disable this option you playlist can be processed foreever.

4 *loop modules:* if the music file ends it will start again. The next file will be played after pressing Enter. If you turn off this option the playlist will play all modules without any user interaction.

5 *scan module information:* When entering a directory the files are processed to gather module information which can be shown. If you disable this option directories will be processed quicker.

6 *scan module information files:* the module information cache in the home directory of OPENCP will be read if this option is enabled.

7 *scan archive contents:* to save hard disk space you can store your files inside archives like `ARJ` or `ZIP`. If the fileselector finds an archive it will open it to scan for files.

8 *scan module information in archives:* if modules are found inside archives they will be decrunched to find any module information. This option can take several minutes if many modules are stored in archives

9 *save module information to disk:* toggles weather to save gathered informations in the module information chache.

A *edit window:* If you don't want the module information window at the bottom disable this option. The directory and playlist windows will spawn over the complete screen.

B *module type colors:* different file types are shown in different colors on the screen. When watched on monochrome monitors or laptops you might want to disable this option.

C *module information display mode:* changes the contents of the directory window. You can also use ALT+tab or ALT+i inside the fileselector.

D *put archives:* Show archives, so they can be accessed like directories. Normally this should be disabled if archives are scanned automatically.

The screen size can be changed by pressing ALT+z. In 132 columns mode some additional module information can be shown in the directory window. If the fileselector is busy scanning the current directory for files, you can interrupt the scanning with ALT+s.

You can delete a file with ALT+k. You will be prompted if you really want to delete this file. When pressing y the file will be deleted. If the file is stored inside an archive the file will be deleted from the archive.[2] The current file can be moved by ALT+m. You will have to type the new path in the path window into which the file will be moved. If the file is inside an archive it will be extracted. You can specify an existing `.ARJ` file and the file will be packed into the archive.[3]

The module information shown in the module information window can be saved to a portable ascii file (see page 55). You may want this feature if you are a composer of music and want to trade your music together with already processed module information files. Start the fileselector and edit the information for the file. Then switch back to the directory window and

[2]If only one file was stored inside a `.ZIP` archive pkzip will leave an empty archive of 22 bytes on your harddisk. See section 5 on how to avoid this.

[3]this works only with ARJ archives by now.

press ALT+w. The fileselector will save a file with the extension `.MDZ` and the filename of the selected file, which stores all module information seen in the module information window. If a directory is scanned and the fileselector finds such `.MDZ` files they will be read and processed. The module information for all files in the current directory can be saved with ALT+a. You have to type the filename manually in the path window without extension!

You may want to change the entry *type* in the module information window if you have old amiga modules or a non-standard midi file. Very old Noise- and SoundTracker modules only had 15 instruments and no file identification. So the fileselector is not able to detect those files as valid modules and refuses to play them. You have to insert `M15` in the *type* entry. If the module does not differ between tempo and speed and is of the 15 instrument type insert `M15t`. Some ProTracker modules do not differ between tempo and speed too. If you have one of those modules use `MODt`. A module player for PC called DMP introduced a feature called panning. To enable this (non-standard) feature insert `MODd`. If you want to play midi files with a second drum track on channel 16 use the `MIDd` option. Any other file should be autodetected correctly. If you have renamed a module to a different extension (say `hello.mod` to `hello.s3m`) OPENCP will refuse to play it, because the file type is wrong. You could correct this by inserting the right file type in the module information as shown above. But it is recommended to rename the file to the right extension instead of tweaking the autodetetion of the player.

The current playlist can be saved into the .PLS format by pressing ALT+p. You have to type the filename without extension in the path window. A standard extension `.PLS` is appended. The playlist can be loaded just like any other module from the fileselector or at startup.

The drive `@:` is a special device which can be used to change the hardware configuration without leaving the player. If you access this drive you will see two subdirectories.

In the `INPUTS` subdirectory you can choose the device which will be used when sampling from external sources (when playing CD audio tracks or starting OPENCP in sample mode). The `DEVICES` directory displays all devices which where detected at startup. Normally you might want to change this if you want to save the next file as a `.WAV` or `.MP2` file to the harddisk.[4] If you have a soundcard with hardware mixing support (Gravis, AWE, EWS) you can enable softwaremixing.

If you are the lucky owner of a soundcard capable of hardware sample playing[5] you only have limited sample memory. If the sample data of the music is larger than the available memory OPENCP will try to reduce the sample size by applying the following steps:

1. Convert 16bit of 8bit samples. This is indicated by a “!” in the bit entry of the instrument section.
2. Half the size of an 8bit sample. This is shown by a small $\frac{1}{2}$.
3. Quarter the size of an 8bit sample, indicated by a $\frac{1}{4}$.

Before *downsampling* the sample data OPENCP will search for the sample with the lowest frequency spectrum. This sample will be converted first. If enough space is freed up OPENCP will stop downsampling. If not the loader will continue until the sample data fits into the sound card memory.

The above behaviour can be avoided if a file is marked *big* by pressing ALT+b in the fileselector. The filesize will turn red. Now a file will not be loaded into the soundcard memory, but played with the internal mixing routines. This limits the size of files only to the size of physical memory.[6]

[4] See section 4.5.
[5] Gravis Ultrasounds, Sound Blaster AWE series and Terratec EWS series
[6] You have to apply a valid *playback-* and *mix-device* to use this feature. See section Configuring on page 27 for details.

3.4. Reference

a. . . z	– quickfind
ALT+a	– write module information `.mdz` for directory
ALT+b	– mark module “big”
ALT+c	– configure fileselector
ALT+d	– goto DOS
ALT+i	– change display mode for directory window
ALT+k	– delete file
ALT+m	– move file
ALT+s	– stop scanning module information
ALT+w	– write module information `.mdz` for selected file
ALT+z	– toggle screen mode
↑, ↓	– move cursor one entry up/down
CTRL+{↑, ↓}	– move module up/down on playlist
→, Ins	– add file to playlist
←, Del	– remove file from playlist
CTRL+{→, Ins}	– add all files to playlist
CTRL+{←, Del}	– clear playlist
Pgup, Pgdown	– move cursor one page up/down
CTRL+{Pgup, Pgdown}	– move module one page up/down in playlist
Home, End	– move cursor to top/bottom of the list
CTRL+{Home, End}	– move module to top/bottom of playlist
Enter	– play selected file – change to directory/archive/drive – edit entry (in module info window)
CTRL+Enter	– edit path window
Tab	– change between directory and playlist
ALT+Tab	– same as ALT+i
SHIFT+Tab	– change to module info window
Esc	– exit fileselector

Supported filetypes – valid options for the *type* entry in the module information window.

669	669 Composer module
AMS	Velvet Studio module
BPA	Death Ralley archive
CDA	compact disk CD audio track
DMF	X Tracker module
IT	Impulse Tracker module
MDL	Digi Tracker module
MID	standard midi file
MIDd	standard midi file, channel 16 is a second drum track
MOD	amiga ProTracker 1.1b module
MODt	amiga ProTracker 1.1b module, effect Fxx is tempo
MODd	amiga ProTracker 1.1b module with effect 8xx is panning
MODf	pc Fast Tracker II .mod file
M15	amiga NoiseTracker module with 15 instruments (plays like ProTracker 1.1b)
M15t	amiga NoiseTracker module with 15 instruments, effect Fxx is tempo (plays like ProTracker 1.1b)
MP3	MPEG audio format level 1-3
MTM	Multi Tracker module
MXM	Mxmplay module
OKT	Oktalyzer module
PLS	OPENCP playlist, works also with M3U and PLT playlist files
PTM	Poly Tracker module
S3M	Sream Tracker 3 module
SID	PSID sid file

UMX	Unreal module file
ULT	Ultra Tracker module
WAV	Microsoft RIFF wave file
WOW	WOW Tracker module
XM	Fast Tracker 2 module

Chapter 4

Player

When OPENCP is started with a valid filename the file is loaded and the player interface is started. This is the main part of OPENCP and you have various ways to display all kinds of music information and data.

4.1. General

OPENCP displays general (status) information in the first 4 rows. Some of these entries can be changed by the user, others are static for each module. The following list will explain every entry.[1]

vol	The bar display the current playback volume. The default value is 100% or full volume. To change the volume press F2 and F3. This will change the volume by one dot each time you press a key. The keys + and - on the numeric keypad will change the volume smoother.
srnd	Toggle this option with F4 to enable a simple surround effect.
pan	If a voice should be played on the right speaker you can rearrange the stereo panning with this entry and F5,F6. To get mono sound adjust the two *riders* to the middle.
bal	Just like the device on your stereo this option works. Press F7,F8 to adjust the stereo balance between full left and full right.
spd	The playback speed can be changed with F9,F10. [2]
ptch	The pitch of the file can be altered with F11,F12
row	Most files of the module type are divided into rows and patterns. The first number shows the currently played row. The second number shows the total number of rows in the current pattern. All numbers are shown in hexadecimal format.
ord	Modules are divided into several orders consisting of patterns. The first number shows the currently played order. The second number shows the total number of orders.[3] All Numbers are shown in hex.
tempo	the current tempo of the file.
bpm	These are not the physical beats per minute, but rather the speed at which the file is played (only valid for module types). This option is often refered by Trackers as BPM.
gvol	Some file formats allow a global volume to be set.
amp	This option lets you adjust an amplification. You can adjust this value with CTRL+F2, CTRL+F3.[4]
filter	You can select different types of interpolation by pressing Backspace:

off	no interpolation
AOI	OPENCP tries to determine if interpolation is neccessary for each note and sample indepentantly. This is the default option and should be enabled.
FOI	every sample is always interpolated. This option uses more processor power as AOI.

[1] Some special file types modify the appearance of the general field, but only static data is affected.

[2] By default the speed and pitch options are linked (indicated by a small ↔). To disable linkage press CTRL+F12. Note that this will not work with all supported file types.

[3] Not all orders have to be played, as modules can jump between different orders.

[4] This is not the same as Volume and F2, F3, because you are able to make the complete file louder than 100% with this option. Note that setting to values above 100% might harm sound quality.

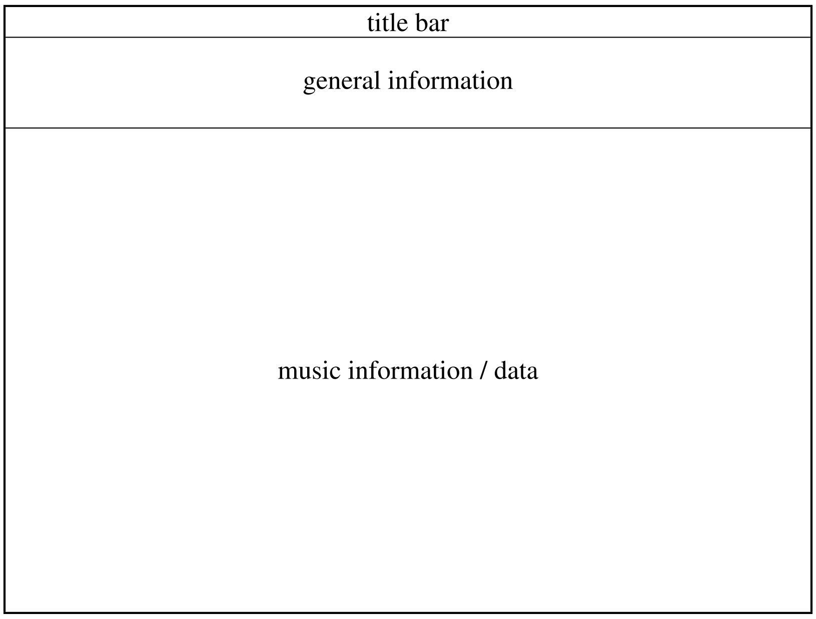


Figure 4.1.: screen layout in the player

module	shows the filename currently played and the title of the file
time	time since starting the current file

4.2. Global functions

Below the general information is a dark grey line. On the left side of the line the current screen mode is shown. In the middle a list of channels. Each file type has a maximum number of channels played simultaneously. For example simple file formats as `.WAV` can have one or two channels (responding to a mono or stereo sample). An audio CD always has 2 channels (left and right). Module file types can have many channels typically ranging from 4 to 32 channels.

The currently selected channel is displayed in light grey. To select another channel use ←,→. You can also use ↑,↓ which will loop through the channels if the left or right end is reached.

Pressing q will *quiet* the selected channel. This key is valid in every part of the player. To enable the channel press q a second time. The reverse logic can be accomplished with s. This disables all other channels than the selected, so only one channel plays solo. Another hit on s will *unsolo* the channel again, so playing all channels. You can use any combination of the above keys. An example: Select channel 1 and press s. Now you will hear only channel 1. Go with the cursor keys to channel 3 and press q. As the channel is currently turned of (quiet) is it now played again. so you hear channels 1 and 3. Now switch to channel 2 and press s. Now only channel 2 will be played, whiche channel 1 and 3 are turned off again. By pressing s again all channels are enabled.

You can directly *solo* the first 10 channels by pressing keys 1. . . 0. This will act as if you had changed to the appropriate channel and pressed s. Channels 11-20 are accessed through ALT+1. . . ALT+0.

To pause the file press p.

The current file can be restarted by pressing CTRL+Home. To move a bit forward use CTRL+→. If a module is played this will skip the current order and start playing the next order. In other file types this command skips a certain amount of time, depending on the estimated playing time. CTRL+← will rewind the music. This is not possible for all file type (midi or sid files). When playing modules the current order is skipped and the previous order is playing from the beginning. To skip a smaller amount of the file use CTRL+↑ and CTRL+↓. This will skip 8 rows when playing modules. If the files support jump or loop command using these functions can lead you to patterns not included in the original play order! Be aware that using these funtions can lead to somewhat crashed files.[5]

The next file in the playlist can be loaded with Enter. If no more files are left in the playlist the fileselector will be started so you can choose the next files. However the current module will continue playing. By pressing Esc you can switch back to the player again. The fileselector can also be invoked with f. The current playlist is shown and can be edited. When exiting

[5]This does not mean that OPENCP itself crashed, but that the order of the music file can be disturbed so heavily, that the player is not able to play the correct music anymore.

the fileselector with Esc you can load the next module in the playlist with Enter. Leaving the fileselector with Enter will load the currently selected module and switching back to the player.

By default a module is looped after its end was reached. You can change this behaviour by configuring the file selector[6] or with CTRL+l. When looping is disabled the next module in the playlist will be loaded once a module has ended. If no modules are left in the playlist the fileselector is started.

A DOS-Shell will be started when typing d. Note that OpenCP uses about 200KB so not all programs might start or execute properly. However it should be enough to make simple tasks like copying, deleting etc. When running under Windows95 the free amount of memory should be the usual size.

The screen mode can be changed by pressing z. This will toggle between 25 and 50 rows textmode, while CTRL+z toggle between 30 and 60 row textmode. ALT+z will toggle between 80 and 132 column mode.[7]

The current configuration can be "saved" with ALT+F2. It is not stored permanently on hard disk, but only valid until OpenCP is quit. Normally the default configuration is loaded every time a new file is loaded. This function makes the current configuration the new default configuration. A previously saved configuration can be loaded with ALT+F3. The *real* default values are loaded with ALT+F4. To modify the default values permanently you have to change the `cp.ini` file as explained on page 27.

An online help is shown by h, ? or F1. Use Pgup and Pgdown to scroll through this screen.

Users of SoundBlasterAWE, Terratex EWS and Gravis Ultrasound PnP can use CTRL+F5 and CTRL+F6 to adjust the onboard reverb effect. The SBAWE can also add a chorus effect which is controlled via CTRL+F7 and CTRL+F8.

4.3. Text mode functions

The player has two different operating modes. Text mode and graphics mode. In text mode you can enable various functions at once, while in graphics mode only a single function can be shown.

Because there can be more than one text mode funtion visible at the same time you might have to press the according key more than once to get the wanted effect. Each function can be in one of the following states:

- invisible - inactive
- invisible - active
- visible - inactive
- visible - active

When pressing a key the according funtion is made active, but left invisible. By pressing the same key a second time the function will be made visible. An active function can be recognized by their title string displayed in bright blue, while inactive functions have their title string displayed in dark blue. Keys affecting the funtions are only processed for the currently active mode. So it might be necessary to change to the appropriate mode by pressing its key once to manipulate its behaviour.

The textmode functions divide the screen into fields shown in figure 4.2. In the 132 column mode only one of the instruments* and channels* fields is active and used by the appropriate function. If a function is not visible the space is used by the other visible functions.

[6] see section 3.3 for details

[7] 80x25 and 80x50 textmode are available on every VGA card. All other modes require a proper VESA bios (at least version 1.2) and may not be available on all cards.

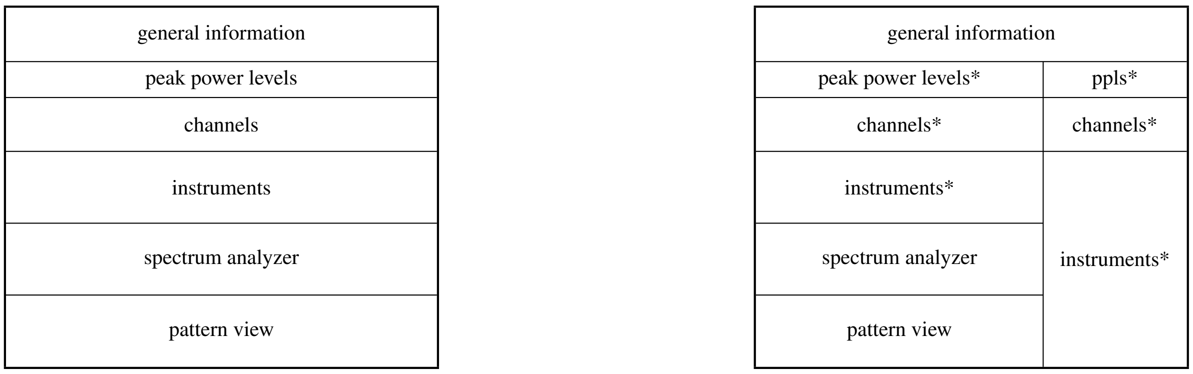


Figure 4.2.: screen arrangements in 80 and 132 column text mode

Channels

The channel function is invoked with c. The channels appear in two different modes. By default the short mode is enabled. Two channels are shown in one row. A grey number shows the channel number. Left to it a white number shows the currently played instrument / sample on this channel followed by the note. If the note starts to play it is shown in cyan for a short while. The third number shows the current volume at which the intrument / sample is played. Behind the volume the current effect is shown.[8] At the rightmost of each entry the current (physical) volume splitted among left and right output channel is displayed in a bar graph.

The currently selected channel is indicated by a small white > to the left side of the channel number. When a channel is muted with s or q it is shown in dark grey. However OPENCP continues to play this channel, so that the music sounds correctly when turning on this channel again.

When pressing c twice the channel function switches to the long format. Each channel now uses a single row as more information is beeing displayed. From left to right this is as follows: channel number, instrument / sample name, current note, instrument / sample volume, pan position, current, volume. [9]

If the textmode is changed to 132 column mode the channel function can be displayed in short form at the upper right corner of the free space as shwown in figure 4.2.

If there are more channels than space inside the screen area OPENCP will scroll automatically through the channel list when you use the cursor keys. This is indicated by white ↑ and ↓ characters.

The channel function has no title string which could indicate if it is active or inactive. So you might have to press c one time more often if the channel function was previously inactive.

Instruments

If the current file a module (or midi) the used instruments / samples are shown with the instruments function. The instruments are shown with i. Just like channels instruments come in two formats, short and long.

In the short view only the intrument names are shown giving you space for two instruments per row in 80 column mode (4 instruments are shown in 132 column mode). An instrument / sample that is currently played is shown in bright cyan. If the sample is played ont the currently selected channel it is shown in bright green. All inactive intruments are drawn dark grey. If a sample has been played once a rectangular dot is placed left to the intrument number.

When the intruments are switched to long mode various information is displayed. From left to right this is as follows:

- a number from 00h to FFh giving the instrument number
- instrument name
- sample number (when using multiple samples per instrument)
- sample name (only in 132 column mode)
- length of the sample in bytes
- length of the loop in bytes
- bits per sample
- the base note. For modules the default is C-4
- finetune value
- standard volume at which the sample is played
- standard pan position
- various flags (volume, pan envelopes etc.)
- fadeout value (only in 132 column mode)

Often a file includes more instruments than can be shown on the screen. Use Pgup,Pgdown to scroll through the instruments. If the instrument function is active CTRL+{Pgup, Pgdown} will scroll for a complete page. When inactive you can scroll single lines by using CTRL+{Pgup,Pgdown}. This is very useful if you have enabled more than one textmode function.

The instrument flags (the rectangular dots left to the instrument number) are cleared with ALT+i. By pressing Tab you can toggle between the color mode and pure grey.

[8]All these informations are only shown when a module or similar type of file is played. The nature of file like `.WAV` or `.MP3` permits those displays.

[9]This layout is only valid for module type files. Other file types like `.SID` have a different layout, but basically showing the same information.

Pattern view

Modules are arranged in patterns. You can view these patterns with the pattern view function envoked with t. When enabling this funtion OpenCP tries to display all channels at once using the best display possible. For modules using few channels (<8) this default display is normally acceptable, but you might want to change it when playing modules with many channels.

The pattern is shown in different columns. At the leftmost the row number is shown in hex. If the screen mode and pattern view allows the row number is shown again at the right side of the screen. Then follow some fields for global commands the module might contain. The biggest section of the screen use the channel columns, each one displaying on single channel indicated by the number on top of the column. Inside such a channel column various information can be displayed depending on the amount of space available. You can see the format of a channel column in the status line of the pattern view. The format of the column can be changed by pressing Tab. As there are many combinations of screen mode, channels and formats I will not go into detail here.

The number of channel rows displayed at once can be changed by pressing Pgup, Pgdown. Normally the pattern view will follow the music as it progresses. With Space the pattern view will stop. The current play position is now displayed with a white ▷ char. You can now browse through the module with Pgup,Pgdown. Space will enable the follow mode again, bringing the pattern view to the current play position.

The pattern view displays the different effects used in modules with different colors. Green is used for effects affecting the pitch of the sample, while blue command change the volume. Effects drawn in purple change the pan position. Red colors indicate the manipulition of the timeslice effected with this samples. Other effects are drawn white. Please have a look inside the online help h, ? or F1 for a complete reference on the effects shown.

Spectrum Analyzer

The spectrum analyzer uses the fast fourier transformation to gather information on the audio spectrum used in sample data. The analyzer is started with a. This function splits the sound data into many *bands* of pure sine waves. This is called the spectrum of the sample.

The status bar of this function shows you the range each bar covers and the highest frequency processed (the rightmost bar corresbonds to this frequency). Use Pgup, Pgdown to change the range. Home will set the default value of 2756Hz.[10]

With ALT+a the mode of the spectrum function can be toggled. Stereo using two analyzers, mono using only one and a single mode are available. In the single mode the currently selected channel is used as sound source for the analyzer.

Tab changes the color used for the analyzer.

To get a smooth view the default charset of the graphics adapter is modified. This is working fine with every graphics card under MS-DOS. However Windows95 does not allow the manipulition of characters, so the spectrum analyzer function is not shown properly when running under Windows95 (window-mode only). If you see a distorted screen while using the spectrum analyzer function this is normal.

peak power levels

This function shows the current physical volume of the output channels in a bar graph. You can use v to make this function visible - invisible. In the 132 column mode the levels can also be shown at the right side of the screen.

Volume control

Since OpenCP 2.5.1 there exists a volume control panel (currently SoundBlaster family only). You can browse through the different items with ↑ and ↓. If you want to change a value, try ← and →. You can also toggle between a short mode, a long mode (only in 132 column modes) and invisible mode with m (Volume control is disabled in 80 column modes and enabled in 132 column modes by default).

Module message

Some file types store messages which can be viewed with ALT+F9 like in Multi Tracker. If the message is long use Pgup, Pgdown to scroll.

[10] The highest possible frequency is half the output frequency (22KHz when playing at 44Khz).

eXtended mode

All four text mode functions can be displayed simultaneously. This function enables channel, instrument, spectrum analyzer, pattern view and volume control function with a good preset in text mode. x will enable 132 column mode. ALT+x will switch to the default 80x25 mode with channel and instrument functions enabled.[11]

4.4. Graphic mode functions

The default graphics mode is 640x480x256. Only one graphics mode function can be shown at once. The screen therefore splits into the general window at the top side showing the usual informatin and the function window covering the rest of the screen. When running under Windows95 the graphics mode can not be displayed in a window. So there may be excessive mode switches if you switch between graphics and textmodes, because windows will try to display the textmode screen in a window. Use the full screen option of Windows95 by pressing ALT+Enter.

If you have included a background picture in the `cp.ini` it will be shown in the graphics modes (expect the graphical spectrum analyzer).

Oscilloscopes

The oscilloscopes are started with o and come in 4 different modes: logical (the channels are sorted with the default panning position), physical (channels 1 to n from top to bottom), master (the mixed output channel(s)) and single (the currently selected channel is shown).

By pressing Tab you can enable/disable triggering of the scopes. If the output is triggered a wave on the screen always starts with the upper halvwave. If triggering is turned off the wave will be drawn from the current position.

The scale of the scopes can be altered with Pgup, Pgdown.

Note dots

n starts the note dots function. Each channel is displayed on a horizontal row. The current note is represented by a dot or bar. Low notes are placed on the left side. High notes appear on the right side of the screen. By pressing Pgup, Pgdown the scale of the rows can be changed. However the default scale fits the usual note scale of modules exactly, so there should be no need to change.

By pressing n you can alter the output appearance of the dots. In the modes *stereo note cones* and *stereo note dots* the current pan position is indicated by the left / right half of the icon.

Graphical Spectrum Analyzer

The graphical spectrum function works in two video modes. By pressing g you will see the standard 640x480 mode. SHIFT+g will start the spectrum in 1024x768 mode. Apart from this difference the two video modes are equal.

Pressing g more than once toggles between the usual stereo, mono and single channel mode for calculating and showing the spectrum. Pgup and Pgdown adjust the frequency range. Home will set the frequency to 2756Hz. To half the resolution (and yet speed up the calculation) press ALT+g.

Tab change the palette of the graphical spectrum. SHIFT+Tab do the same for the standard spectrum analyzer at the bottom.

If you have difficulties interpreting this function here is a short explanation. The standard spectrum analyzer at the bottom shows you the frequency spectrum at the current moment. The higher a single bar, the louder the frequency. Now imagine looking at this spectrum from top, now every bar becomes a single dot. The height of the bar is now coded into different colors (from black $\leftrightarrow$ low to yellow $\leftrightarrow$ high). Now we can draw these point along the screen and see the spectrum as is progresses over time. This is somewhat a “3D” view of the spectrum, with the frequency coded along the y-axis, intensity coded in different colors and the time along the x-axis.

[11] If your VESA bios does not support 132 columns a 80x50 mode is used.

Phase graphs

The last graphical function is started with (b). You can toggle between four modes which correspond exactly to those in the oscilloscope mode. This function displays the currently played samples in a phase graph. One full wave of the sample is drawn over the complete angle of a circle. The louder the sample the greater the radius of the circle. A sine sample would respond to a normal circle.

Würfel mode

With (w) the würfel mode is enabled. It's only purpose is to display an animation located in the home directory of OPENCP. The (Tab) key will change the play direction. To save diskspace no animations are included in the distribution of OPENCP. They can be found on the OPENCP homepage (page 37). Animations can be generated with the wap program from bitmap files.[12]

4.5. Using the diskwriter and MP3writer

OPENCP can write all sound output directly to hard disk. Data is written in standard `.wav` format. You can use this feature to burn audio cds from any sound format supported by OPENCP.

Although you can write `.wav` files in every possible sample format you should not alter the default of 44100KHz, 16bit, stereo. You should disable module looping so each module is written once onto harddisk. You can disable looping in the `cp.ini` with the directive `loop=on` in the `[fileselector]` section. You can also change looping temporarily with the fileselector configuration while OPENCP is running. See the fileselector advance usage page 11 for details.

To enable the diskwriter device you can change the `cp.ini` file and select the `devpDisk` as default device by moving it to the start of the `playerdevices` directive in the `[sound]` section. You can also select the `devpDisk` device temporarily by selecting the `@:\DEVICES\DEVPDISK.DEV` device.

Now simply start OPENCPand select a module to play. You will hear no output and notice that the module is played with maximum speed[13]. In the directory where you have started OPENCP(not necessaryly the directory where the module is located) subsequent `.wav` files named `CPOUT000.WAV`, `CPOUT001.WAV` will be created.

To save modules as MP3 files the same steps apply, but you have to select the `devpMPx` device in the `cp.ini` file or use the `@:\DEVICES\DEVPMPX.DEV` special device. Bitrate and CBR/VBR mode can be altered in the `[devpMPx]` section.

4.6. Using the Compo mode

If you enable the *Compo mode* in the `cp.ini` file all title and instrument string from modules will not be displayed.

4.7. MIDI files

OPENCP is able to play MIDI files. However there is a certain problem. Unlike the other file formats MIDI does not store the sample information needed to produce a sound output. The midi file only contains which instrument out of a set of 127[14] should play which note at a given time. This is the reason why `.MID` files are much smaller than other file types.

This has of course some disadvantages. To hear a MIDI file you need to have some information how to play the used instruments. Back in the old days the OPL2 sound chip which was present on the SoundBlaster cards was used to play the midi instruments. Most people find the sound capabilites of the OPL series rather limited and midi files were no big deal back then.

Things changed when so called wavetable cards became popular. Those card have sample data stored onboard in a ROM plus a hardware mixer capable of mixing several midi channels. The MPU-401 interface from Roland is the de facto standard for accessing those cards, but this feature is not supported by OPENCP yet.[15] A disadvantage is that those wavetable cards only have a very limited memory for sample data typically 4MB. If you imagine 127 instruments and 64 drums fitting into just 4MB you can guess what sound quality these cards have. Modern cards have normally much more onboard ROM/RAM, but in our opinion even 32MB are far too less for a good sound quality.

OPENCP goes a different way. Instead of using the onboard ROM samples, the samples needed for a specific `.MID` file are loaded on demand from the harddisk into main memory and then processed by either the hardware mixer (if you have such a

[12] See appendix 11.
[13] depending on your cpu power
[14] a set of drums is defined aswell
[15] however this might change in the near future

card and the samples fit into its memory) or by the software mixers. This has the advantage that you can easily change a single instrument, if you don't like the default sound.

The instruments are stored in so called *GUS-patches*, a file format introduced by Gravis with their UltraSound cards. At first you have to get a complete patch set. If you own a GUS classic or GUS max you will probably have those files already on your harddisk. All instruments are stored in files ending with `.PAT`.

You can use the original GUS patches (available on our homepage), look for the EAW patches on the net or try the freepats.

If you unpack any of those archives to a folder on your harddrive, you have to adjust the path in the `cp.ini` file. The configuratoin directive

```
ultradir=c:\gus\midi   ; path to ULTRASND.INI
```

should be set to the folder where you extracted the patch files. Please check that a file called `ultrasnd.ini` is located in this folder, as OPENCP needs this file, to load the patch files.

4.8. running OPENCP on small computers

OPENCP runs on every MS-DOS system that supports 32bit protected mode. However you have to configure a bit to get OPENCP running, because those old computers mostly lack MHz and ram.

config.sys and autoexec.bat

Apart from `himem.sys` no other drivers are needed to run OPENCP. Therefore you should modify the startup files to provide maximum memory. Especially you should disable all disk caches and ram drives. Below is are two sample files you could start from.

The `config.sys` should look like:

```
device=c:\dos\himem.sys /testmem:off
```

The `autoexec.bat` should look like:

```
set blaster=a220 i7 d1
set ultrasnd=220,5,5,7,7
```

You can load additional drivers (mouse, cdrom) if you like. Most drivers only use dos memory below 1MB and therefore do not *steal* memory above 1MB that OPENCP has to use. But please do not load programs such as `smartdrv` or `ramdrive` as they use XMS/EMS memory, which will then not be available anymore.[16]

The `set` statement for the sound card installed should be present in the `autotexec.bat` so OPENCP knows what sound cards to search for.

virtual memory

OPENCP usually needs more than 4MB memory. If you have 4MB or less you can still use the player if you enable the virtual memory functionality of the `dos4gw.exe` protected mode extender. Just as windows it can swap your memory to hard disk and thus provide more memory to applications than physical memory is present.

To enable use of virtual memory simply place the following line in your `autoexec.bat` or type it at the command line:

```
set dos4gvm=1
```

Mixer

If your computer is too slow to play with proper speed remember that the new Float Mixer is the default device used by OPENCP when dealing with software mixing. If you enable the Normal Mixer you will gain a good speed up of your system.

Look in the `[sound]` section of your `cp.ini` file for the following line:

```
wavetabledevices=devwMixF devwMixQ (...)  devwMix devwNone
```

The leftmost device is used as default. So change the line to the following to enable the Normal Mixer:

```
wavetabledevices=devwMix devwMixF (...)  devwNone
```

If you don't understand all this right now, read chapter 5 on how to configure OPENCP.

[16] A nice documentation about the startup files is Configuring your MS-DOS properly.

Interpolation

If the player still runs to slow you can disable the use of interpolation with software mixing. Look for the following line in the `[sound]` section of the configuration file:

```
filter=1
```

and change it to:

```
filter=0
```

Now the use of interpolation is disabled. You can enable the filters again in the player with backspace.

still to slow?

If you applied the above 4 tips and OpenCP is still running too slow, there's hardly anything left to tune. Remember that graphic modes are generally slower than text modes. And in text mode the analyzer uses most ressources. If you only display channels, instruments and track list there's almost no cpu consumption by visuals.

If the player is still too slow your last chance is to lower the mixing / playing rate of the player. Locate the following line in the `[sound]` section of `cp.ini`:

```
mixrate=44100
```

Use the table 4.3 as a guideline to set this value.

Figure 4.3.: command mixing rates

44100	CD Quality
33000	very close to CD
22050	Radio Quality
11025	Telefon Quality
8000	`.au` Quality

While applying those patches please remember that modules with more channels will *always* need more cpu power than those with few. If your Impulse Tracker modules (`.it`) always click and pop while old Amiga modules (`.mod`) play fine that's normal, because the modern trackers allow more than 4 channels.

4.9. Key Reference

ESC – quit the player
F1 – help
F2, F3 – volume up/down
CTRL+{F2, F3} – change amplification
ALT+F2 – *save* current configuration
ALT+F3 – load previously saved configuration
F4 – surround on/off
ALT+F4 – load default configuration
F5, F6 – change panning
CTRL+{F5, F6} – adjust reverb
F7, F8 – change balance
CTRL+{F7, F8} – adjust chorus
F9, F10 – change speed
ALT+F9 – song message
F11, F12 – change pitch
F11 – toggle between 6581 and 8580 (sidplayer only)
F12 – toggle between PAL and NTSC (sidplayer only)
CTRL+F12 – (un)lock speed and pitch
1. . .0 – solo channel 1. . .10
ALT+1. . .0 – solo channel 11. . .20
CTRL+1. . .0 – solo channel 21. . .30
a – textmode spectrum analyzer
ALT+a – toggle analyzer mode
b – phase graphs
c – channel mode
d – goto DOS
f – goto fileselector
g – graphic spectrum analyzer
SHIFT+g – graphic spectrum analyzer in 1024x768
ALT+g – toggle fast/fine algorithm
h – help
i – instrument mode
CTRL+i – instrument mode colors on/off
ALT+i – remove *played* dots
CTRL+j – same as Enter
CTRL+l – song looping on/off
ALT+l – pattern looping on/off
m – volume control
CTRL+m – same as Enter
n – note dots
o – oscilloscopes mode
ALT+o – behaves like Tab in this mode
p – pause
ALT+p – pause screen
q – quiet current channel
s – solo current channel
t – track/pattern mode
CTRL+ü – same as ESC
v – peak power level mode
w – würfel mode
x – eXtended mode
ALT+x – normal mode
z – toggle between 25/50 or 30/60 rows
CTRL+z – change between 25/50 and 30/60 rows

ALT+z – change between 80 and 132 column mode
Enter – play next song in playlist
Space – stop pattern mode flow
Pause – pause screen output
Backspace – toggle filter
Tab – change option of the activated mode
’ – link view
° – make screenshot
,, . – fine panning
+, - – fine volume
*, / – fine balance
→, ←, ↑, ↓ – change current channel
CTRL+→ – skip the current pattern
CTRL+← – restart current pattern / goto previous pattern
CTRL+↓ – skip 8 rows
CTRL+↑ – skip -8 rows
Ins – goto fileselector
Pgup, Pgdown – scroll in current window
CTRL+{Pgup, Pgdown} – scroll in instruments window (eXtended mode)
Home, End – goto top/bottom of current window
CTRL+Home – restart song

Chapter 5

Configuration

OPENCP can be widely configured using the `cp.ini` file, which should reside in the home directory of the player itself. This file is processed at every startup and informs the player about various modules / plugins and their respective configuration. For average users the default configuration should be sufficient. However if you want to change certain aspects of the player permanently you have to modify this file. You can use every ascii editor to edit the ini file.

The `cp.ini` consists of many sections. Each section starts with a section identifier tag in square brackets [ ]. Examples of sections are `[general]` or `[sound]`. The sections id tag should be followed by a newline. After a section has been declared with the id tag the options describing the sections follow. Each options takes a single line and consists of a keyword followed by a "=" and the parameters. Example: `mixrate=44100`.

All options following a section id are options for that section. If a new section starts with a id tag all forthcoming options are assigned to the new section. All options for a section have to be grouped together. Multiple declarations of sections are not valid.

```
[general]
  link=dos4gfix
[defaultconfig]
  link=mchasm
[general]
  dos4gfix=off
```

the above example has to be written as:

```
[general]
  link=dos4gfix
  dos4gfix=off
[defaultconfig]
  link=mchasm
```

Note that in the above example the option *link* has not been overridden by the *defaultconfig* section. Both sections now can access an options named *link*, but both options are totally independant of each other.

Comments can be placed anywhere in the configuration file and are marked by a `;`. The rest of the line starting from the `;` is considered as a comment and not processed.

Normally the definition for an option ends with the end of the line. If many parameters are needed to specify an option they may exceed the default line width of 80 characters. Although this is no problem for OPENCP it is not nice looking. You can extend a line logically by using the unix like *backquote* at the end of a line to begin a newline without interupting the current option definition.

```
[example]
  option1=parameter1 parameter2
  option2=parameter1 \
          parameter2
```

Both options contain exactly the same parameters.

When modifying the configuration you should always start with the default configuration file and configure it to your needs. Building a bug free config file from scratch is difficult.[1]

We will now have a look at the individual sections and their options.

5.1. general

The *general* section describes which internal modules or plugins to load at startup. Most of them are required for normal operation of OpenCP so you should not remove any of the contents. All options listed in this section are loaded every time OpenCP starts! The default *general* section looks like:

```
[general]
  link=dos4gfix mmcmphlp poutput hardware inflate pfilesel \
       cpiface fstypes
  ultradir=c:\sys\iw\
  dos4gfix=off
;  datapath=
;  tempdir=
```

link	this options describes the modules to load when starting OpenCP. There is no need to change this option, unless you have coded a basic internal module.
ultradir	If you want to hear MIDI music you need samples for the general midi instrument set.[2] This option contains the path to the midi patches. Section 4.7 describes how to set up OpenCP for midi playback.
ados4gfix	DOS4GW is a protected mode extender used by the watcom c++ compiler. However this extender has problems with IRQ greater than 7. If you use the default dos extender `cplaunch` you should disable this option. If you want to use DOS4GW for whatever reason and your soundcard uses an IRQ greater than 7 you have to enable this.
datapath	OpenCP searches for background pictures and animations in its home directory. If you want to store your artwork at a different place use this option to set the right directory.
tempdir	this directory is used for extracting modules from archives. If you have set a DOS environment variable called either `TEMP` or `TMP` these will be used.[3]

5.2. defaultconfig

The *defaultconfig* section is very similar to the *general* section. But unlike the *general* section which is always processed the settings in the *defaultconfig* section can be ommited with an alternative section and the `-c` flag from the command line. If the `-c` flag is not present the *defaultconfig* section will be processed.[4]

```
[defaultconfig] ; default configuration
  link=mchasm devi sets smpbase plrbase mcpbase arcarj arczip \
       arcrar arcumx arcbpa arclha arcace playcda playinp
  prelink= ; preloaded dlls
```

link	just like in the *general* section this option defines which modules should be loaded at startup. You can delete some entries if you will not need them – however this is not recommended as they do not use much memory and do not require any processor power.
prelink	these files will be loaded before starting the main module. If something goes wrong here OpenCP will continue to work.

5.3. sound

This section is the most important one for using OpenCP. If you want to change the configuration permanently you have to modify the entries of this section.

[1] And remember to make backups before changing vital parts of the `cp.ini`

[2] a so called midi patch set

[3] Most programs make use of those environments, so you should set them at startup in your `autoexec.bat` file.

[4] Therefore it was named *defaultconfig*...

```
[sound] ; default sound section
  playerdevices=devpWSS devpEWS devpSB devpESS devpGUS devpPAS \
                devpNone devpDisk devpMPx
  samplerdevices=devsWSS devsSB devsGUS devsNone
  wavetabledevices=devwMixQ devwIW devwGUS devwSB32 devwDGUS \
                   devwMix devwNone
  ampegtomono=off
  ampegmaxrate=48000
  mixrate=44100
  mixprocrate=4096000
  mix16bit=on
  mixstereo=on
  plrbufsize=500
  mixbufsize=500
  samprate=44100
  samp16bit=on
  sampstereo=on
  smpbufsize=2000
  defplayer=
  defsampler=
  defwavetable=
  midichan=64
  itchan=64
  cdsamplelinein=off
  bigmodules=devwMixQ
  wavetostereo=1
  waveratetolerance=50
  amplify=100
  panning=100
  volume=100
  balance=0
  reverb=0
  chorus=0
  surround=off
  filter=1
```

playerdevices	OpenCP uses three different devices to communicate with the hardware. The *playerdevices* are used to play a stream of samples. As all sound cards support this feature you will find *playerdevices* for every sound card supported by OpenCP. This device is needed for playing `.wav` and `.mp3` files and if you want/have to use the software (quality) mixer. OpenCP searches for all devices listed in this option at startup and only those found are actually loaded. You can delete all devices you have not installed to speed up to startup procedure. If you have multiple sound cards installed be sure to list all devices if you want to use more than one sound card.[5]. If more than one device is listed the first in the list will be used as default.
samplerdevices	these devices are the least important ones. They are only needed if you want to use OpenCP when playing cd audio tracks or start the player in sample mode. The sample data is not calculated from files, but sampled from either the cd, line or microphone jack of the sound card. You can then use the graphical screens to view the sounds.
wavetabledevices	for mixing several channels you have to use *wavetabledevices*. Most sound cards are only able to play to channels simultaneously normally assigned to the left and right channel or your home stereo. The *mixer* devices are used to mix the sample data of module files to those two channels. However modern sound cards have special hardware to mix channels “onboard”. But all hardware mixers have a maximum amount of channels to mix[6]. Especially `.it` files often use more than 32 channels so an errorfree playback can not be guaranteed when using hardware mixing. You should include one of the software mixers for this case.
ampegtomono	play `.mp3` files only mono. This might be necessary if you don’t have the latest generation of Intel processors.
ampegmaxrate	the highest playback sample rate to use for `.mp3` files.

[5]you can change devices by using the special `@:` drive described in section 3.2

[6]mostly 32 channels

mixrate	the default mixrate. Unless you have a very old sound card[7] or a very old processor[8] there is no need to change this option.
mixprocrate	if you have a slow cpu[9] you might not be able to play 32 channels at full mixrate. This value defines the maximum "calculation power" to which OPENCP tries to use the full mixrate.
mix16bit	Leave this option enabled unless you have a 8bit sound card.
mixstereo	dito for stereo cards
plrbufsize	When using a *playerdevice* to play sample streams a DMA buffer is used to minimize cpu resources for handling the sample stream. This option sets the DMA buffer length in miliseconds.
mixbufsize	When running in a multitasking environment there is no guarantee for constant cpu resources. To avoid a break in the sample stream OPENCP will calculate in advance. This option sets the buffer lenth in miliseconds.
samprate	When using a *samplerdevice* this value will be used.
samp16bit	dito
sampstereo	dito
smpbufsize	this options sets the length of the sample DMA buffer in miliseconds.
defplayer	with this option you can override the default *playerdevice*. Normally you don't need to set this option, as the default device can also be set by the order in the *playerdevice* option. This option can also be specified by using the `-sp` options from command line.
defwavetable	this option sets the default *wavetabledevice*. Can also be set with `-sw` command.
defsampler	the same as the `-ss` command line option.
midichan	unlike the module file types the MIDI file format does not specify the exact amount of channles to use. Complex `.mid` files can easily try to play several dozens of channels simultaniously. This option sets the maximum amount of channels to mix when playing `.mid` files. Note that this value has no effect when a hardware mixing device is used, as the maximum number is limited by the hardware.
itchan	the `.it` format can play more than one sample per channel simultaniously. A maximum number of channels to mix is required for this file type too. When playing `.it` files using a hardware mixer the maximum number of channels is again limited to the hardware.
cdsamplelinein	If you select a `.cda` file the cd input of your sound card is used to sample the current music. If you do not have a cd input or if you have connected your cd-rom to the line-in jack enable this option to change to sample input.
bigmodules	This option is of interest for users of hardware mixing devices only. Sound cards capable of mixing channels are not only limited by the amount of channels played simultaniously, but by the amount of onboard memory to store the sample data too. If files are marked as "big" in the fileselector this device listed in this option will be used for mixing this module.[10]
wavetostereo	When playing a mono wave the sound card can either be switched to mono mode or the wave can be played as a stereo file.[11]
waveratetolerance	if the sample rate of the wave file is not equal to a sample rate supported by your sound card, OPENCP will not resample unless this value is exceeded. Divide the value by 1000 to get the percentage.
amplify	the default amplification to use within the player. The following options are described in section 4.1 in detail. The command line option `-va` overrides this option.
panning	the default panning (command `-vp`)
volume	the default volume (command `-vv`)
balance	the default balance (command `-vb`)
reverb	some sound cards have an onboard effect processor[12] which features a reverb effect. This option controls the intensity of the onboard effect. (command `-vr`)
chorus	dito as reverb (command `-vc`)
surround	this options controls the fake surround effect[13] (command `-vs`)
filter	The software mixer can use a software filter to enhance the playback quality. Different algorithms can be used. (command `-vf`)

0	no filter
1	AOI - only filter samples when filtering is necessary
2	FOI - filter every sample even if filtering has no effect

[7]SoundBlaster 1.x or SoundBlaster Pro and compatibles
[8]Something like 386SX
[9]<486DX
[10]See section 3.3 on page 12 for details about this feature
[11]switching the soundcard may cause problems so enable this option.
[12]currently the SoundBlasterAWE and the TerraTecEWS
[13]this has little to do with real Dolby Surround although there should be a certain effect if you have such an amplifier

5.4. screen

When the player starts it will use the options in this section as the initial appearance.

```
[screen]
  usepics=opencp01.tga
  compomode=off
  startupmode=text
  screentype=2
  analyser=on
  mvoltype=1
  pattern=on
  insttype=0
  channeltype=1
  palette=0 1 2 3 4 5 6 7 8 9 a b c d e f
```

usepics	When using graphics modes you can use a picture to show in the background. OPENCP will load any TGA files with 640x384 dimensions and 256 colors. As the TGA format is poorly implemented in modern graphic programs this might change in the future. As some colors out of the 256 are used by OPENCP you should leave either the first or the last 16 colors in the palette black. The pictures should be copied to the home directory of OPENCP, unless you specify a different location in the *defaultconfig* section.
compomode	this option will enable the compo mode. Section 4.6 describes the details.
startupmode	start the player in either text or graphic mode
screentype	the default screentextmode: 0 – 80x25 1 – 80x30 2 – 80x50 3 – 80x60 4 – 132x25 5 – 132x30 6 – 132x50 7 – 132x60
analyzer	if the player starts in textmode show the analyzer (or not)
mvoltype	the appearance of the peak power levels: 1 – big 2 – small
pattern	show the tracklist when starting OPENCP in textmode
insttype	the appearance of the instrument function: 0 – short 1 – long 2 – side (only in 132 column modes)
channeltype	the appearance of the channels in textmode: 0 – short 1 – long 2 – side (only in 132 column mode)
palette	with this options you can redefine the default colors used in textmode. The first entry defines which color to use for the original color with number 0. Leave things as they are if you are satisfied with the visual appearance of OPENCP. We will provide new color schemes in the future.

5.5. fileselector

Except the first two options all options can also be specified at runtime by pressing ALT+z in the fileselector. However to make changes permanent you have to modify the *fileselector* section.

```
[fileselector] ; default fileselector section
  modextensions=MOD S3M XM IT MDL DMF ULT AMS MTM 669 NST WOW \
                OKT PTM MXM MID WAV RMI MP1 MP2 MP3 SID DAT \
                PLS M3U PLT
  movepath=  ; default path to move files
  screentype=2
  typecolors=on
  editwin=on
  writeinfo=on
  scanmdz=on
  scaninarcs=on
  scanmnodinfo=on
  scanarchives=on
  putarchives=on
  playonce=on
  randomplay=on
  loop=on
  path=.
```

modextensions	files containing these extensions will be scanned by the fileselector. Only those files will be shown. If you want to load files with different extensions you have to specify them at the command line.[14]
movepath	the standard path to move files into. This is describend in section 3.3.
screentype	the textmode to use within the fileselector. The options are the same as in the *screen* section.
typecolors	show files in different colors depending on the file type
editwin	show the edit window at the bottom of the screen
writeinfo	write the info to the information database located in the home directory of OPENCP. This speeds up the processing of directories, as files have to be scanned only once.
scanmdz	if `.mdz` files are found in the current directory, they will be scanned and the included information used.
scanmodinfo	scan inside the music files for module information.
scanarchives	if archives (like `.zip` or `.rar`) are found in the current directory the are scanned for modules.
putarchives	show archives in the fileselector, so they can be used just like subdirectories.
playonce	play every file only once (thus not looping it) and then procede with the next file in the playlist. If the file contains a loop command the loop command is ignored.
randomplay	play files in the playlist in random order.
loop	loop files after the end.
path	the default path to use when starting the fileselector the first time. The default is the current directory (.). If you keep all your music files in one directory you can specfiy this directory here.

5.6. device configuration

The following sections define the various devices for the player. Unless you really know what to do you should not change the following options. As most entries are similar only some educational examples are listed here. For a complete reference have a look at your personal `cp.ini` file for details.

The general form of a device looks like:

```
[handle]
  link=...
  subtype=...
  port=...
  port2=...
  irq=...
  irq2=...
  dma=...
  dma2=...
  bypass=...
  keep=...
```

handle	The internal name to use within the player. The `cp.ini` file must contain all *handles* listed in the devices options of the *devices* section.
link	the name of the dll function this device will be linked to.
port(2)	the primary/secondary port of the sound card. This value has to be given in hexadecimal with preceeding *0x* or appending *h*!
irq(2)	the primary/secondary IRQ of the sound card

[14]however files with different extensions are likely to be no valid module format, so they will be refused to load

dma(2)	the primary/secondary DMA channel of the card
bypass	skip the autodetection if it may encounter problems
keep	keep the driver resident in memory, even it is not currently needed.

Most device functions of the standard dll contain autodetection routines for the supported sound cards, so there is normally no need to specify any of the port, irq or dma options. However if OPENCP is not able to detect your sound cards settings you can try to insert the appropriate values in the configuration file.

The next subsections will look at the special features the different sound cards and drivers support. The original order of the `cp.ini` has been slightly modified for the purpose of documentation.

Sound Blaster

Most compatibles should work with the following sections and options aswell. Be sure to include a mixing device aswell if you want to play module file types.

```
[devpSB]
  link=devpsb
  sbrevstereo=off
; subtype= 1:sb 1.x, 2:sb 2.x, 3:sb pro, 4:sb16
[devsSB]
  link=devssb
  sbrevstereo=off
```

sbrevstereo	some revisions of the Sound Blaster Pro (and clones) have a bug, when playing stereo. The right and left channel are played on the appropriate other channel. To get the right order enable this option.
subtype	if your card is not properly detected (this can happen when clones are not 100% compatible) you can set the type of card with this option 1 – SoundBlaster 1.x mono 22Khz 8bit 2 – SoundBlaster 2.x mono 44Khz 8bit 3 – SoundBlasterPro either mono 44Khz 8bit – or stereo 22KHz 8bit 4 – SoundBlaster16 stereo 44KHz 16bit

Terratec EWS

```
[devpEWS]
  link=devpews
  reverb=75
  surround=10
```

reverb	the onboard effect processor will use this amount of reverb on the ouput.
surround	dito

When starting OPENCP in a Window95 DOS-Box those settings will not work, because Windows refuses to set up the effect processor. You have to use the Windows control panel to change the default values set up at windows startup.

Windows Sound System

Sound cards using the Crystal Codec are normally Windows Sound Systems compatible. The Gravis Ultrasound Max series has this codec aswell, but located at a non-standard port. You might have to specify the soundcard type manually.

```
[devpWSS]
  link=devpwss
  wss64000=off
; subtype=
[devsWSS]
  link=devswss
  wss64000=off
```

wss64000	as the Crystal Analog Codec can use sample rates up to 64Khz you should enable this feature to get crystal clear sound playback.
subtype	if your sound card is not detected automatically use this options to set it: 0 – Windows Sound System 1 – Gravis Ultrasound with 16bit Daughter board 2 – Gravis Ultrasound Max

Gravis Ultrasound

The Gravis Ultrasound series divides into two subseries. The old standard includes the Gravis Ultrasound (with daughterboard), Gravis Ultrasound Max and Gravix Ultrasound ACE which use the old GF1 chip for hardware mixing. The new Gravix Ultrasound PnP use the AMD Interwave chip for hardware playback.

```
[devwGUS]
  link=devwgus
  gusFastUpload=on
  gusGUSTimer=on
```

gusFastUpload	use a non-standard feature to upload the sample data to the fast ram. If you experience problems with corrupted playback, or if your system locks when loading a file disable this option.
gusGUSTimer	use the internal GUS timer for looping and synchronization. This enables more precise playback, but may cause problems.

If you own two GUS cards of the old series you can use a special feature of OPENCP using both sound cards to enable quadruple or 3D playback.[15]

```
[devwDGUS]
  link=devwdgus
;  port2=250
; subtype=
```

port2	the base port of the secondary GUS, if it was not autodetected.
subtype	you can specify how OPENCP will use the second GUS: 0 the second GUS will play the rear stereo channels. This is true quadrophony. Of course you have to have a second stereo and a second pair of boxes. 1 this will enable 3D playback. The additional boxes have to placed exactly behind the audioence, on the floor and the other at the ceiling.

OPENCP will still use the panning information provided by the music files and place the channels randomly in the remaining dimensions. If you have access to a second GUS be sure to check this feature one day, as the sound experience is really amazing.

For users of the new GUS PnP with the Interwave chip the following section is relevant.[16]

```
[devwIW]
  link=devwiw
  iwIWTimer=off
  iwEffects=on
  iwForceEffects=off
```

iwIWTimer	use the internal sound card timer for synchronization. When using OPENCP under DOS there is no need to use this feature. However if you want to use the player under Window95 you should enable this to prevent timer inconsenties from windows.
iwEffects	enable the onboard effects (reverb). This feature uses 64KB of the soundcard memory and 4 channels. So the maximum amount of channels is reduced to 28. If the music uses more channels OPENCP will turn off the effect to use the last channels or memory.
iwForceEffects	if you enable this option the effects will always be present, thus overriding the above option.

[15]This is a real 3D playback and has nothing in common with those fakes like QSound etc.
[16]The driver should work with other sound cards using the interwave chip aswell. However we have not encountered such card by now.

MPx writer

OpenCP is able to store the sample data directly to mpeg compressed audio streams. At the moment only MP1 and MP2 are supported, but this might change in the near future.

```
[devpMPx]
  link=devpmpx
  bitrate=128000
```

bitrate	set the mpeg audio bitrate/s.

software mixers

By default OpenCP will use its own routines for mixing several channels to the two stereo output channels. You have the choice between to mixers. The normal mixer is faster in calculating, thus can mix more channels at the same time. The quality mixer however produces better sound ouput. For average modules and a pentium processor the quality mixer should be fast enough for sufficient playback. If many channels are used you may have to change back to the normal mixer[17]

Both mixers take identical options. As the mixers will be rewritten in the future the options are likely to change. Therefore they are not documented here. Please have a look at future versions of this document if you want to change to mixer settings. However these devices never have caused any trouble/bugs and there should be no need for change.

5.7. archivers

When processing files inside archives OpenCP has to know how to call the programs. The following sections define the usage of different archivers supported by the OpenCP fileselector.

When substituting the command line 4 special variables are processed by the fileselector for accessing the archives:

%%	evaluates to “%”
%a	evaluates to the archiv name preceeded by the path to process.
%n	evaluates to the filename of the file to process inside the archive.
%d	evaluates to the destination directory used to extract files out of archive.

Three default command are defined for the arc modules:

get	this command is used when a file should be extracted out of an archive.
put	if a file should be added to the archive this command line is used.
delete	with ALT+k you can delete files out of archives.

The following two archivers contain special options described below:

```
[arcZIP]
  get=pkunzip %a %d %n
  put=pkzip %a %n
  delete=pkzip -d %a %n
  deleteempty=on
```

deleteempty	the program pkzip/pkunzip used by OpenCP to process `.zip` files have a bug when deleting the last file inside an archive. An empty archive of 22 bytes is left on the hard disk. With this option enabled the fileselector will delete those “zip zombies”.

```
[arcACE]
  get=ace32 e %a %n %d
  scaninsolid=false
```

scaninsolid	scan in solid archives. As this takes more time you can disable this feature.

[17] You can toggle by using the *bigmodule* feature described in section 3.3.

5.8. filetypes

The `cp.ini` file contains descriptions for all supported file types. These features will be included in the file loader devices in the next version of OPENCP, so these options will soon be obsolete. There should be no need to modify any of the file types.

Chapter 6

Support

The official internet homepage of OPENCP can be found under:

`http://www.cubic.org/player`

Please send any suggestions and bug reports via electronic mail to Dirk Jagdmann doj@cubic.org.

Many people complain about music files not being played properly (especially with module formats). Please keep in mind that the original tracking/sequencing program is *always* the reference for correct playback. If a music file sound different compared to another player, please keep in mind that perhaps the other program does not play it correct. This is especially true for Amiga 4channel modules. As those files can be produced using various trackers (Noise-, Sound-, Protracker and many, many more) and sadly all of those trackers play slightly different, a general player like OPENCP can not play *all* modules correctly. We try to emulate the behaviour of ProTracker 1.1b, so that most modules are played correctly. However certain features of one tracker permit another feature of a different player.

Chapter 7

Frequently Asked Questions

this chapter was initially written by Mom/Cubic.

Q1: *What is* OpenCP*?*
A1: OpenCP is a music player which plays a variety of sound formats on several sound cards.

Q2: *What sound cards are supported ?*
A2: Gravis UltraSound / MAX / DaughterBoard / PnP
SoundBlaster 1.x / 2.x / Pro / 16 / SB 32 / PnP / AWE / 64
WSS compatible cards
Pro Audio Spectrum series
ESS AudioDrive 688
Terratec EWS64L/XL
Disk Writer, writes .WAV output on disk.
MP3 Writer
Quiet Player

Q3: *What music formats are supported?*
A3: MOD/NST/WOW, XM, S3M, DMF, MTM, ULT, 669, OKT, PTM, AMS, MDL, IT, MIDI, SID, and MPEG 1 audio layer 1/2/3

Q4: *Which is the last version of* OpenCP*?*
A4: The last OpenCP version is the 2.6pre6 Version.

Q5: *Can not find* `dos4gw.exe`*?*
A5: download it: dos4gw.exe

Q6: *Where has* `cplaunch.exe` *gone?*
A6: It was killed, as it sucked. It caused severe timing errors with Wave table cards, as `pmode/w` seems to have some problems with Interrupts. Anyway, if you are in SUCH a need of it, you'll surely find someone who stil has it

Q7: *Does* OpenCP *run with Windows 3.11?*
A7: **not perfect!** If you try it, your system might crash !

Q8: *The sound stops/hangs when I try to switch* OpenCP *to background in Win9x - how can I avoid it?*
A8: If you have a GUS or Interwave and use Hardware mixing, try to enable the soundcard's timer (RTFM please) - if you use software mixing, note that background playing is only available at the main player screen for some technical reasons. If it doesn't work THEN - well, go complain to your soundcard's driver programmer.

Q9: *Always when I'm in "X" mode (132 columns) and I try to switch back to a window,* OpenCP *crashes! Why?*

A9: It's not OpenCP which crashes, it's Windows. Sadly, most graphic card drivers get confused when you try to switch a 132 column mode to a window - and hang the whole system. There's nothing that can be done about it, except that you don't try this again. Sorry.

Q10: *I have an SB16 and use* OpenCP *under windows - why does the playing stop whenever I shell to DOS and try to get back?*

A10: Actually, we have no idea. It seems that the SB16 Windows drivers modifies some of the card's registers in this moment which confuses either the playing or the DOS shell port monitoring routines of OpenCP.

Q11: *Now, how does that MIDI player thing work?*

A11: OpenCP's MIDI player does not, like other programs, use the MIDI capabilities of your sound card / Korg Trinity or whatever, but uses the patch files of the famous Gravis Ultrasound to synthesize the MIDI output itself (just like Timidity or kmidi under KDE) via it's normal wavetable system. So the only thing you need are the GUS patch files which are available at our site. Simply open a new subdir called ”MIDI” in OpenCP's directory, copy all patch files `.PAT` into it and there you go.

Users of a Gravis Ultrasound only have to make sure their `ultradir` environment variable is set correctly or state the patch file and `ultrasnd.ini` location in `cp.ini`.

Q12: *Does that mean I can listen to MIDI files even with my old SBPro?*

A12: Of course - if your soundcard works with the rest of OpenCP, you will be able to play MIDI files, too - as long you as got the GUS Patch files as written above.

Q13: *Hey, I have a module which* OpenCP *plays wrong! You SUCK!*

A13: Well, it's in fact YOU who sucks, as you didn't send that module together with a bug report to doj@cubic.org. How the hell can we fix bugs if we don't know them?

Q14: OpenCP *plays my* `.it` *files much too soft, IT's internal mixer sounds a lot better, why don't you just fix it?*

A14: Because this is a religious question. OpenCP's mixing routines are designed for maximum sound quality which also includes things like declicking and quadratic interpolation, IT's mixing routines are just optimized for speed without really caring about sound quality (speaking of the non-MMX routines). So OpenCP's sound differs a bit, if you are not pleased with it, feel free to use IT again ;)

Q15: *Can you give me the source or a part of the source from* OpenCP*?*

A15: The complete source code can be found on the official OpenCP sites, covered under the GNU GPL.

Q16: *I have a problem finding a small routine that will play mod files. I wanted to know if you can help finding a routine, lib or something like this.*

A16: Try our Tiny XM Player, a free XM Player inc. source.

Q17: *Why don't you implement Nibbles, or similar game, in Cubic? Like FT2-nibbles? Or just plain Snake...*

A17: Get ahold of a coder, give him the source and force him to implement it, if it's a GOOD nibbles version, we might include it into the ”official” release ;)

Q18: *I have a AWE32 and a GUS in my computer. Will both cards be supported simultaneously in the nearby future?*

A18: Right now, only 2 GUSs' are supported simultaneously.

Q19: *What does ”Würfel Mode” actually do?*

A19: ”Würfel” is german word for cube/dice. The mode plays an animation.

Q20: *Can I make my own ”Würfel Mode” animation?*

A20: Of course you can. On our site you'll find the program WAP (WürfelAnimator Professional) which can convert `.pcx` frames to a Würfel Mode animation.

Q21: *How can I put a picture to the background on* OpenCP*?*

A21: Convert your Picture to a TGA-file, these pics have to be 640x384x208 TGAs, the first or the last 48 colors must not be used. You can also save your image as a 640x384x208 gif87 picture. Either the first or the last 48 colors of the palette have to be black.

Q22: *Up to version 2.0alpha++,* OPENCP *ran fine on my 486, now it creeps and can only mix a hand full of channels. Can I fix this somehow?*

A22: RTFM ;) No, as we're living in the age of Pentium computers now, we set the Floating Point Mixer as default. Just change this.

Q23: *I have an AWE or GUS card and the displays are not correctly synced to the music.*

A23: If you have an AWE or GUS OPENCPwill probably use the hardware mixing drivers. The display system of OPENCPis designed to work correctly with software mixing, as it always calculates with a certain buffer delay, caused be the software mixers. Now the hardware mixing cards don't use this buffer and therefore the display is delayed by a certain amount of time. Perhaps some day a mighty coder might fix this.

7.1. Windows Vapc driver specific

The following are errors messages that might appear in the cphost log along with a short description what caused them and how to avoid them.

The DLL `vocp.dll` couldn't be loaded the `vocp.dll` wasn't found. be sure that it's loadable, that means somewhere in your path. `windows\system` should be a good place

WARNING: Couldn't set priority class! *or* WARNING: Couldn't set thread priority! the host couldn't instruct windows to give him more cpu-power. there will be serious skipping while playing.

ERROR: Couldn't load `VAPC.VXD`! Is the `VAPC.VXD` in your texttwindows"system-directory? yeah, the `vapc.vxd` must be in your system directory.

DEVPDX5: couldn't lock primary ... *or* DEVPDX5: couldn't play on primary ... no problem, the driver will use secondary buffer.

DEVPDX5: couldn't create directsound-object your soundcard is in use by some other stuff, maybe a dosbox, maybe *this* dosbox. remove all other hardware-devices from your *cp.ini*, as described in the beginning of this document.

If you own a sound card supported by OPENCP and have difficulties using the vapc drivers there is the chance that both drivers interfere. You may try to remove all native OPENCP drivers from the ini file. The corresponing lines should look like this:

```
playerdevices=devpVXD devpNone devpDisk devpMPx
samplerdevices=devsNone
wavetabledevices=devwMixF devwmixQ devwMix devwNone
```

The native-driver will use the soundcard, and windows will reserve the device for the OPENCP-task.[1] If this occurs, you will find something like *couldn't create dsound-object* in the cphost-log. so, remove the other device-driver, restart the dosbox, and enjoy.

It *must* be possible to launch some direct-sound-output while the cp is *running* (not neccesarily *playing*). try a winamp or something first (using DirectSound, *not* WaveOut). If this works, continue using OPENCP with DirectSound.

When switching to the fileselector playback get interrupted even on very fast computers. Threre is no way to avoid this, because under Windows DOS programs get not enough *interrupt ressources*.

Q24: *When I use the vapc driver and change into the fileselector while playing, the playing stutters.*

A24: This is a well known problem. Windows doesn't give OPENCP enough priority for the background playing. Currently there is nothing that can be done about this.

[1] better: OPENCP-VM

Part II.

Developers Manual

Chapter 8

Writing plugins

Picture 8.1 is a schematic of the modules OpenCP is built from. It shows which parts of the player interact with each other.

The capabilities of OpenCP can be extended by plugins. The type of plugin is not limited, so new graphics modes, file types and even enhanced players may be added by the user. As development for OpenCP is done using Wacom C++ 11.0 it is recommended to use this compiler for developing plugins aswell. We have not tested the plugin interface with other compilers.[1]

The following section was written by Felix Domke for the 12th issue of the hugi diskmag.

8.1. Device Development

Introduction to FDOs, DLLs and the CP in general

Of course, no player can play all formats without errors, support all soundcards or open all archive formats. While you would have to wait for a new version of "normal" players, you can do something against it for OpenCP by yourself.

From the technical point of view this is possible since CP 1.666. Here, the so-called ḞDOs were introduced, "flatmode dynamic link objects". All in all these are nothing different than DLLs which everyone knows from Windows, OS/2 or also Unix. FDOs contain eġdrivers for soundcards. If a new soundcard was to be supported, not a new Cubic Player would have to be released but only a new FDO.

With an entry in `cp.ini`, you can embedd such a driver.

In Cubic Player 2, these FDOs were replaced by DLLs, iė in general only a different format (however, I am not sure whether FDO also supported imports.)

The format FDO had been invented by pascal, which means that it was not supported by any linker. A different thing is the case with the DLLs of CP2. They are simply normal LE-DLLs. Everybody who has done a little more with Watcom C and DOS4G or PMode/W should know the LE-format: It is used there by default.

In general, there is no great difference between LE-DLLs and LE-EXEs. LE-EXEs, however, usually contain no imports or exports (the right professional-version of DOS4GW, DOS4G, supports DLLs).

To reach that the player also works with the normal DOS4GW or even PMode/W, OpenCP contains a DLL-loader which loads these DLLs and links them to the main program. The main program merely consists only of this DLL-loader (later more about this).

Everyone who has Watcom C can create LE-DLLs. Of course it's also possible with other linkers and compilers, but you ought to use Watcom.

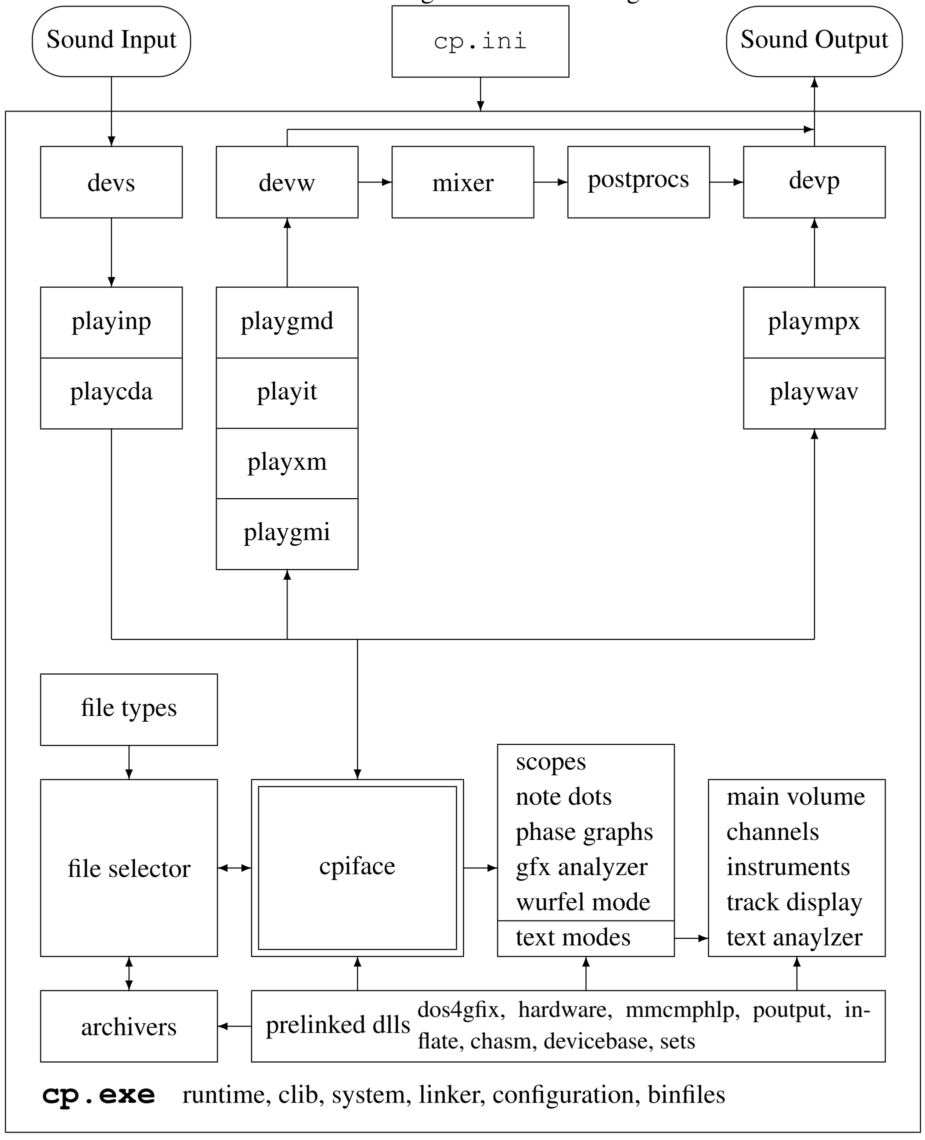
How to create DLLs

If you have already created DLLs once or more, you can probably ignore this chapter.

Otherwise, it isn't so hard either. Everyone of you has probably made a program with more than one object file. The various object files were linked to a single large EXE in the end.

[1] This might change in the future.

Figure 8.1.: block diagram of OpenCP



With DLLs, you usually don't have to change *a single line* of code.

You link one or more object files to a DLL. If other DLLs access to symbols[2] in a DLL, these symbols have to be *exported* from the DLL which contains them and *imported* in the DLL which uses these symbols.

The linker does that by itself. You just have to tell it where what symbols can be found.

For this purpose, *import libraries* exist. They are similar to normal libraries apart from the fact that they don't contain normal code but just links to the DLL. You handle these import libraries like normal libraries (and object files), you just link them to the rest. When the linker needs some function which is described in the library it creates an import (iė a relocation/fixup entry) the target is the corresponding DLL and the corresponding symbol name).

You have to export by EXPs, which are files that contain a list with the to-be-extracted symbols. On linking you pass them to the linker which then creates exports.[3]

EXPs just look like the following to export the function `printf` and the variable `i`.

```
'printf_'
'_i'
```

You create import libraries by `WLIB x.LIB +x.DLL` presumed that `x.DLL` contains exports.

The main EXE is actually just such a DLL apart from the fact that it doesn't contain any imports but an entry-point to `main_`.

the Internals

All DLLs of OPENCP are in `cp.pak`, which is a Quake-compatible `.pak` file. You can unpack it with normal Quake tools.

I roughly describe the most important DLLs which are in this `.pak` file:

`arc*.dll`	archive-reader (e.g. ARCZIP.DLL for .ZIP-files)
`cpiface.dll`	the interface (the screens with trackview and so on)
`devi.dll`	auxilary routine for the devices system that reads from `cp.ini` (*) what devices are to be loaded and loads them
`devp*.dll`	player-devices, play samples
`devs*.dll`	sampler-devices, are more or less able to sample (e.g. for displaying the FFTs of Audio CDs)
`devw*.dll`	wavetable-devices, play more samples at the same time, either directly on the hardware (GUS, SB32...) or mix them and play them on DEVPs (DEVWMIX, DEVWMIXQ)
`fstypes.dll`	module-detection, song-infos for various module-formats are evaluated here
`hardware.dll`	routines for bending and using IRQs, the timer and DMAs so that not every device driver needs its own functions
`load*.dll`	module-loaders, load various formats
`pfilesel.dll`	the file selector
`play*.dll`	plays modules on various devices
`(*)`	Since version 2.0 many things are not directly retrieved from `cp.ini` but from an exported `_dllinfo`. Read more about this later.

A simplified overview about OPENCP:

`cp.exe` loads important DLLs, starts DEVI to get the desired devices, and starts the file selector. This program part asks the user which module he wants, seeks the corresponding LOADer, loads the module, seeks the corresponding PLAYer, and passes the module to it. The player seeks the corresponding devices and passes the data that should be played to them.

At the same time the player passes data to the interface, which was initialized some time before. The interface displays these data and evaluates inputs from the user.

Our first Cubic-DLL

To create a DLL, you first have to create an environment where you can do this best. The easiest method is to install the source of OPENCP and put one's DLL into the makefile. However, since the directory gets messy soon, I recommend developing one's DLLs elsewhere. At first you should get all `.lib` and `.h` of OPENCP by compiling the player once.[4]

[2] in this case, symbols are variables and functions

[3] An export is just an entry in the entry-table but with a name.

[4] Soon there is said to come an archive with all `.lib` and `.h` for those people who just want to develop something new.

Let's start it all over from the beginning step by step. As an example, we want to write an ACE-reader which displays the descriptions before the final decompressing.

First depack the environment, set the paths correctly and copy the *TEMPLATE* from the `indev` directory to `indef\arcace`.

There you have to change the makefile, namely in this way:

```
dest = arcACE.dll
 ...
arcace_desc = 'OpenCP .ACE Archive Reader (c) 1998 Felix Domke'
arcace_objs = arcace.obj
arcace_libs = cp.lib pfilesel.lib
arcace_ver = 0.0

arcace.dll: $(dlldeps) arcace.exp $(arcace_objs)
            $(libdir)$(arcace_libs)
            $(makedll)
            copy arcace.dll \opencp\bin
```

You have to enter all destinations for `dest`, iė all DLLs which shall be created in the end.

`arcace_desc` is the description which will be printed.

`arcace_objs` are the object-files which shall be linked to the DLL.

Since there is an implicit-rule which says that WPP386 gets executed for CPPs for which OBJs with the same name are demanded, there also has to be an `arcace.cpp` with our code.

`arcace_libs` are the (import) libraries from which the imports shall be searched. Standard functions like `sprintf_` etc. are exported in `cp.lib`. PFILESEL.LIB contains some functions we will need later.

`arcace_ver` is the version which gets displayed in the linkview of OPENCP. You compute it in a rather strange way. How exactly, I myself haven't understood, but usually you should convert the version from hex to decimal (eġ0x024000 for 2.5.0) and then divide by 100. But someone must have erred.

The `copy` at the end copies the DLL right into the CP directory.

In general in an archive-read, two functions are passed, one to read the file names from the archive and "tell" them to the file selector via callback and another to call the unpacker to unpack a file from the archive.

To be concrete, it looks like the following:

An `adbregstruct` gets exported which looks like this:

```
struct adbregstruct
{
  const char *ext;
  int (*Scan)(const char *path);
  int (*Call)(int act,
              const char *apath,
              const char *file,
              const char *dpath
             );
  adbregstruct *next;
};
```

`ext` is the extension, in our case `.ace`.

`Scan` is a function which gets called with the archive as parameter and tells it to the file selector by calling `adbAdd()`.

`Call` will get executed then to unpack a file from the archive.

We leave out `next` because it will be set at runtime anyway.

Now let's export such a struct:

```
extern "C"
{
adbregstruct adbACEReg = {".ACE", adbACEScan, adbACECall};
char *dllinfo = "arcs _adbACEReg";
};
```

`dllinfo` makes an entry in `cp.ini` superfluous. It provides that CP takes notice of our archive-reader.

However, something like `link=...  arcace.dll` has to be in `cp.ini`.

Our `adbACEScan()` looks like the following[5]:

```
static int adbACEScan(const char *path)
/*
  "path" is the filename of the archive.
  If everything works, 1 will be returned, otherwise 0.
*/
{
 [...]
 sbinfile archive;
/*
  "binfile" is the Library which is used in CP for (almost) all
  file accesses. I think it is EXTREMELY practical. It is
  actually quite self-explanatory. "sbinfile" is a normal file
  on disk.
*/
 if(!archive.open(path, sbinfile::openro))
  return(1);
/*
  When we can't open the archive, we can exit at once :)
*/

 unsigned short arcref;

 char arcname[12];
 char ext[_MAX_EXT];
 char name[_MAX_FNAME];
 _splitpath(path, 0, 0, name, ext);
 fsConvFileName12(arcname, name, ext);
 arcentry a;
 memcpy(a.name, arcname, 12);
 a.size=archive.length();
 a.flags=ADB_ARC;
 if (!adbAdd(a))
 {
  archive.close();
  return 0;
 }
/*
  Here, the archive itself gets added to the file listing,
  namely by adbAdd(arcentry &).
  The struct which adbAdd expects looks like:

  struct arcentry
  {
   unsigned short flags;
   unsigned short parent;
   char name[12];
   unsigned long size;
  };

  "flags" is e.g. ADB_ARC for archives.
  "parent" is not so important at the moment.
  "name" is the file name in 8.3-format (fsConvFileName12
         converts it pretty nicely :)
         (e.g. from "x.y" to "X       .Y  ")
  "size" is simply the length which will be displayed.
*/
```

[5]The function has been *heavily* simplified and can't be run in *this* form, but all things concerning OPENCP archives are in it. Only the ACE-specifical things, most of all the thing with the SOLID-archives, are missing.

```
  arcref=adbFind(arcname);
/*
  Aferwards we keep the "ref" in our mind so that we can enter
  it as "parent" afterwards. parent is the source-archive so
  that the files there will be unpacked from the right archive
  afterwards... (The file selector creates a list of all files
  in the directory and the files in the archives. For the last
  thing, it calls the archive-reader. If the user presses with
  "Enter" on a file afterwards, the file selector has to know
  what ARCer it has to call and most of all from which archive
  the files come. Therefore the parent has to be set.
*/
 [...]

/*
  Now let's deal with the ACE itself. The ACE format is very
  similar to the RAR one, a detailed
  @REF="format description":"coace.txt"

  In any case the function "ReadNextHeader" reads the header on
  the current position and skips additional bytes if they appear
  (i.e with files the packed data). In this way header comes
  after header, at least it appears. (In reality ReadNextHeader
  SEEKs at the beginning, not at the end, but that's not
  important now, don't get confused by that :)
*/

 arcentry a;
 while(ReadNextHeader(archive))
 {
/*
   Here, the header gets read.
*/
  char ext[_MAX_EXT];
  char name[_MAX_FNAME];
  [...]
  if(aheader->type==1)
  {
/*
  ...if the type is 1 (file), the file name will be read...
*/
   [...]
   char filename[_MAX_PATH];
   memcpy(filename, afheader->filename, afheader->filenamesize);
   filename[afheader->filenamesize]=0;
   strupr(filename);
   _splitpath(filename, 0, 0, name, ext);
/*
   ...splitted into "name" and "ext"...
*/
   if(fsIsModule(ext))
/*
   ...and if "ext" is a module-extension (see CP.INI), then...
*/
   {
    a.size=afheader->unpsize;
    a.parent=arcref;
    a.flags=0;
    fsConvFileName12(a.name, name, ext);
/*
   ... this file will be added to the file list by adbAdd.
*/
    if(!adbAdd(a))
    {
     archive.close();
/*
   If the whole thing doesn’t work, well, then it doesn’t work.
*/
     [...]
     return 0;
    } else
    {
/*
   Otherwise, if wished,
*/
     if(fsScanInArc&&[...])
     {
/*
some blocks get be depacked (by the UNACE-routines in UAC_DCPR.*)
*/
      dcpr_init_file();
      int rd=dcpr_adds_blk(buf_wr, size_wrb);
      unsigned short fileref;
      fileref=mdbGetModuleReference(a.name, a.size);
/*
   ...a reference for the just ADDed file will be retrieved...
*/
      if((fileref!=0xFFFF)&&(!mdbInfoRead(fileref)))
      {
       moduleinfostruct ms;
       if(mdbGetModuleInfo(ms, fileref))
       {
        mdbReadMemInfo(ms, (unsigned char*)buf_wr, rd);
/*
  ...and then we e.g. look for the song name etc. by
  "mdbReadMemInfo". Just for explanation, again:

  "buf_wr" contains about the first unpacked 2-4kb of the actual
  file. Now "mdbReadMemInfo" tries to get to the infos in it
  with various, format-specific routines.
*/
        mdbWriteModuleInfo(fileref, ms);
/*
  These will also be set then.

  Another thing about the "mdbInfoRead" and "mdbGetModuleInfo":
  At first you have to read what is already known about the
  file, then actualize it and write it back.
*/
       }
      }
     }
    }
   }
   [...]
  }
 }
 [...]
 archive.close();
 return(1);
}
```

Now let’s come to `adbACECall`. This function is quite easy. Depending on `act`, it has to pack, unpack, move etc. the file *file* from the archive *apath* to *dpath*. To make it easier, only one `act=="adbCallGet"` is supported, iė unpacking.

```
static int adbACECall(int act,
                      const char *apath,
                      const char *file,
                      const char *dpath)
{
 switch (act)
 {
  case adbCallGet:
  {
   return !adbCallArc(cfGetProfileString("arcACE", "get",
                                         "ace e %a %n %d"),
                      apath, file, dpath);
/*
   "adbCallArc" is a nice function: it replaces something like
   %a, %n and %d with the passed arguments. The first argument
   has to be inserted for %a, the second for %n and the third
   for %d. Beforehand, "cfGetProfileString" gets from CP.INI
   whether the user wants to use another packer (XACE etc.) and
   that in the section "arcACE" and the key "get=".
   If the key hasn't been found, "ace e %a %n %d" will be taken
   by default, which, by the way, should be correct almost every
   time. :)
*/
  }
  case adbCallPut:
   // not implemented
   break;
  case adbCallDelete:
   // not implemented
   break;
  case adbCallMoveTo:
   // not implemented
   break;
  case adbCallMoveFrom:
   // not implemented
   break;
  }
  return 0;
}
```

Now a simple `wmake` compiles the whole thing presumed that a working `makefile` is available.

Chapter 9

Compiling OPENCP

To compile OCP you need the following software:

- Watcom C++ 11c Install the compiler suite into a directory of your taste. You then have to set up some environment variables. Use the following batch as a starting point.

  ```
  set watcom=c:\watcom
  set path=%watcom%\binnt;%watcom%\binw;c:\tasm;%path%
  set include=%watcom%\h;%watcom%\h\nt
  set lib=
  ```

  If you want to use your Watcom compilers under plain DOS you have to exchange the `binw` and `binnt` directories.

  Currently the OpenWatcom distributions (including version 1.2) can not compile OPENCP. You have to use the version 11c mentioned above. The build process however needs the program `wstub.exe` which is not included in the 11c archive. If you own the retail version of the Watcom compiler, you can either use your old program file or you can get the file from the OpenWatcom distribution. Please get OpenWatcom 1.2, install it somewhere and copy `wstub.exe` into the `binw` directory of your Watcom 11c tree.
- TASM — the Turbo Assembler from Borland. Versions 4.x and 5.x should work. It is commercial software and still sold.
- NASM — Nasm version 0.98.35 is reported to work. You should either use the windows versions named `nasmw.exe` or the djgpp dos version. Nasm can be found at

Look into the file `makefile.cfg` and adjust paths to nasm, tasm and watcom if necessary. Now you should be able to build everything with a simple ”wmake”. ”wmake install” will copy all the files built into a `bin` directory located in the source tree.

The `wmake` utility accepts some parameters. Only one of the following may be supplied at a time.

all	compile OPENCP(this is like supplying no switch)
install	compile the program and the copy program into the `\bin` directory.
clean	delete all compiled `.obj` files
distclean	delete all files that have been ”made” once. This is like unpacking the source distribution again.

9.1. Windows helper programs

The downloadable version from Watcom C++ currently doesn’t support Win32 compilation. To compile the `cphost.exe` program you need the full version of Watcom. You can then build `cphost.exe` with `wmake cphost.exe`.

The DirectX 9 SDK is required to build `cphost.exe`. However Microsoft ships its SDK with library version 6.0 whereas Watcom only handles version 5.0. You can either use the batch file below to convert the libs (with lib.exe from some MS Visual Studio) or download the libs and headers.

```
lib.exe DxErr8.lib    /convert /link50compat /out:DxErr8.l
lib.exe DxErr9.lib    /convert /link50compat /out:DxErr9.l
lib.exe amstrmid.lib  /convert /link50compat /out:amstrmid.l
```

```
lib.exe d3d8.lib      /convert /link50compat /out:d3d8.l
lib.exe d3d9.lib      /convert /link50compat /out:d3d9.l
lib.exe d3dx.lib      /convert /link50compat /out:d3dx.l
lib.exe d3dx8.lib     /convert /link50compat /out:d3dx8.l
lib.exe d3dx8d.lib    /convert /link50compat /out:d3dx8d.l
lib.exe d3dx8dt.lib   /convert /link50compat /out:d3dx8dt.l
lib.exe d3dx9.lib     /convert /link50compat /out:d3dx9.l
lib.exe d3dx9d.lib    /convert /link50compat /out:d3dx9d.l
lib.exe d3dx9dt.lib   /convert /link50compat /out:d3dx9dt.l
lib.exe d3dxd.lib     /convert /link50compat /out:d3dxd.l
lib.exe d3dxof.lib    /convert /link50compat /out:d3dxof.l
lib.exe ddraw.lib     /convert /link50compat /out:ddraw.l
lib.exe dinput.lib    /convert /link50compat /out:dinput.l
lib.exe dinput8.lib   /convert /link50compat /out:dinput8.l
lib.exe dmoguids.lib  /convert /link50compat /out:dmoguids.l
lib.exe dplayx.lib    /convert /link50compat /out:dplayx.l
lib.exe dsetup.lib    /convert /link50compat /out:dsetup.l
lib.exe dsound.lib    /convert /link50compat /out:dsound.l
lib.exe dxguid.lib    /convert /link50compat /out:dxguid.l
lib.exe dxtrans.lib   /convert /link50compat /out:dxtrans.l
lib.exe encapi.lib    /convert /link50compat /out:encapi.l
lib.exe ksproxy.lib   /convert /link50compat /out:ksproxy.l
lib.exe ksuser.lib    /convert /link50compat /out:ksuser.l
lib.exe msdmo.lib     /convert /link50compat /out:msdmo.l
lib.exe quartz.lib    /convert /link50compat /out:quartz.l
lib.exe strmiids.lib  /convert /link50compat /out:strmiids.l
del *.lib
ren *.l *.lib
```

If you want to modify the VXD driver you need the Win98DDK. But you can not download it anymore from Microsoft. Try a search on Google for the filename 98ddk.exe.

9.2. Compiler switches

In the beginning of the makefile you can choose between release and debug builds. The windows build is broken. You can also add these #defines to the `cflags` in `makefile`.

RASTER	enable a fake raster bar every time the mixer starts. You can then see how much cpu time the mixing routines use.

9.3. wmake

If you experience problems with `wmake` please make sure you have enough dos memory available. You should supply at least 580KB. You can simply check how much memory is available by using `mem /c /p`. This command will tell you the memory usage of each single loaded driver and programm. If you find out that there is not enough dos memory available you can free memory by unloading some cd-rom drivers, mouse drivers or smartdrv etc.

During compilation wmake sometimes exists with “not enough memory” error messages. To continue building just `cd` back to the man directory and start wmake again.

Chapter 10

.MDZ files

This was written by Fabian Giesen, based on the original `.MDZ` description by Niklas Beisert.

10.1. What are `.MDZ` files?

`.MDZ` files are Module Description files (the *DZ* is a short form of *DIZ*, which is in turn a short form of Description). The `.MDZ` format can, however, contain much more information than normal Module/Sound files, which normally only contain the song's title and sometimes a "song message", which can be freely entered by the author. Things like the name of the author of a song, its playtime, a comment about it etc. which can be entered in OPENCP's Fileselector cannot be stored inside the module file, so they have to be stored somewhere else. Usually this is in the file `cpmodnfo.dat`, which normally perfectly suits this task.

However, this format has some disadvantages. Imagine you give a module file to a friend. Simply copying your `cpmodnfo.dat` together with the song on the disk is impractical, because this file overwrites your friend's `cpmodnfo.dat` when he simply copies it. So he could take your description, write it down somewhere, and take it with him, but that is pretty uncomfortable.

That's why the `.MDZ` files were invented. `.MDZ` files contain the description of one or more module file(s), just like the `cpmodnfo.dat`, but have attributes which make them interesting:

- `.MDZ` files are normal text files and therefore human readable
- The content of `.MDZ` files gets automatically inserted in the module description database `cpmodnfo.dat` without need for user interaction
- `.MDZ` files can be exported from within the OPENCP fileselector with one keystroke (see section 3.4).

10.2. Why should I use them?

(This section is intended for composers)

There are of course some good reasons for using them:

- Every OPENCP user sees your description automatically when he moves the cursor over the file
- The module type is also stored in `.MDZ` files, so you can explicitely force OPENCP to play your Module in Protracker (or FastTracker) mode, for example.
- Even if the user does not use OPENCP, he can still read your description, because `.MDZ` is a simple, human readable text format
- Creating `.MDZ` files is almost no extra work (you can create a complete `.MDZ` file from within OPENCP in less than one minute)

So, as you see, using `.MDZ` files has many advantages, but (almost) no disadvantages. It is surely worth trying.

10.3. The `.MDZ` file format

This is not documented here. If you want a complete description, wait for the soon to be released OPENCP Technical Reference Manual.

Chapter 11

WürfelAnimator WAP

This chapter is written by Felix Domke, the author of WAP.

11.1. What is the so called WürfelMode?

In version 1.0 of the famous Cubic Player, pascal got a nice idea of playing silly animations inside the Player.

He invented the *WürfelMode* (*Würfel* is the german word for cube). The WürfelMode was able to play animations with a resolution of 160x100 pixels in 256 colors in a special tweaked vga-mode. The animation format was a special one.

11.2. How can I create such animations by myself?

Long time, only the cubic team (or even only pascal?) itself was able to create such animations. The reason for this was that, as said, the animations had to be in a very special format, and the only tool that was be able to create this format were the not-so-famous *WürfelTools*. Included there was a tool called `makedmp.exe`. It took various `.tga` files, and made a `.dat` animation out of them. But noone checked how this tools worked, mainly because it had no command-line-help and the sources weren't available anymore.

But times are changing. Today, you don't have to guess how to use a utility anymore. You just have to read the documentation (yes, even you, Realtime ;). Today, you can even use a resolution of 320x200 (wow, THAT's highres).

For people who didn't guess how pascal's tool worked, I made a new one. This time *with* source available to all interested people. This time with `.pcx` images, not *.tga* one, because `.pcx` is much easier to read, and also more widely supported. This time even with some palette-aligning, so you don't have to care about same palettes anymore.

11.3. And how can I use your tool?

First, you have to render/paint/rip a animation. It HAS to be 320x200, or, if you want to make an old, < OPENCP 2.0q, animation, 160x100. In case you didn't notice, the OPENCP 2.0q and higher has a new WürfelMode, called *WürfelMode*] [, with support for 320x200 pixels images, instead of the lowres 160x100 ones.

Your animation should be in the form of many, many `.pcx` files, called for example `PIC000.PCX`, `PIC001.PCX`, `PIC002.PCX` and so on. They must have only 256 colors, and they should have all the same palette. If they don't have the same palette, the palette will be aligned. But my calculation of this isn't very good (but it works :), so for BEST results you have to do this in your favourite picture-editing-program. Next, you have to create a *script*. This script contains informations about the filenames of your frames, and a some other stuff.

The script has the following format:

```
Version
[SFO] RLECompression
Title
NumFrames Delay Filename
[NumFrames Delay Filename]
```

```
[NumFrames Delay Filename]
[...                     ]
```

Version is either 0 or 1, 0 for 160x100 animations, 1 for 320x200. If you want to create a Version 0-animation, *SFO* should be set to 1, if you want to create a Version 1-animtion, *SFO* has to be left out. [1]

RLECompression should be set to 1, it compresses the Animation a little bit.

Title is the title of the animation, it shouldn't be longer than 31 chars, it's displayed inside the fileselector of OPENCP.

After this *header*, a random number of section follows (but at least ONE ;).

Every section describes a sequence of some PCXs. In the final animation, all sections are joint together.

NumFrames is the number of frames in this sequence. For example, if you have PCXs that are named from `PIC000.PCX` up to `PIC199.PCX`, NumFrames should be 200.

Delay is handled a little bit different in the two versions. If you want to create a Version 0-animation, a *Delay* of 1 means a framerate of about 21.3 frames per second, a *Delay* of 2 is about 10.65 fps (the half), a *Delay* of 3 is about 7.1 (a third) and so on.

If you want to create a Version 1-animation, *Delay* is 65536/desired fps, so if you want to have 15fps, "Delay" should be 4369. This value has to be BELOW 65536 (and above 0, of course :)

Filename is the filename of your sequence. Because most animations have more than 1 frame, you can use %d (and other C-printf placeholders). For people whose native language is not C, here some short examples:

filename	framenumber	real filename
pic%d.pcx	0	pic0.pcx
	1	pic1.pcx
	2	pic2.pcx
	3	pic3.pcx
	9	pic9.pcx
	10	pic10.pcx
	11	pic11.pcx
pic%03d.pcx	0	pic000.pcx
	1	pic001.pcx
	2	pic002.pcx
	3	pic003.pcx
	9	pic009.pcx
	10	pic010.pcx
	11	pic011.pcx

The framenumber always starts with 0, not with 1, keep this in mind!

Here is an example script: (version 0)

```
0
1 1
a basic cubic-player animation
11 2 intro%d.pcx
200 1 ani%03d.pcx
```

This animation would have 211 frames (`intro0.pcx` to `intro10.pcx`, then `ani000.pcx` to `ani199.pcx`).

And here again one for version 1:

```
1
1
an enhanced OpenCP animation
11 3072 intro%x.pcx
200 1000 ani%03d.pcx
```

To create the animation, just type in
`wap <scriptfilename.scr> <outputfilename.dat>`
and hit Enter.

The WürfelAnimator will now try to make the animation. To view the animation inside OPENCP, rename the created `.dat`-file to `cpani001.dat` (or `cpani002.dat` and so on...) and press w inside the player.

Anyway, maybe you have some fun with this tool.

[1] If you're interested what this value means, well, just get the original documentation for professionals, it's described there

Part III.

Appendix

Appendix

Thanks

The OPENCP maintainers thank the following people for supporting us in the development of the player and its associated tools:

- Thomas Carey
- Jacob Poon
- Igor Shterv
- Michael Schwendt
- Moritz Grimm
- Maximilian Rehkops
- Luciano Golino
- Oleg Prokhorov
- Vladimir Sojka
- Pham Dank Quang
- Evzen Polenka
- Bertolt Meyer
- Lukas Grunwald
- Michael Baeckow

Without you OPENCP would not have the current face and features. We also like to thank the hundreds of unnamed people supplying us with bug reports and general suggestions concerning the player. Please keep on using and supporting OPENCP!

Appendix

Soundcard Setup

This chapter summarizes initialization of various soundcards under DOS.

.1. SoundBlaster

SoundBlaster cards need a environment variable `BLASTER`. To set it issue a command similar to the following. Note, that you only need the A, I, D, T and possible H parameter.

```
  SET BLASTER=A220 I5 D1 H6 T6 P330 E620
```

A	Base port of the SoundBlaster. Common values are 220, 240 and 260.
I	Interrupt of SoundBlaster. Common values are 5, 7 and 11.
D	8bit DMA channel. Common values are 1 and 3.
H	16bit DMA channel. Common values are 5, 6 and 7. Only used by T¿=6.
T	Type of SoundBlaster. 1. SoundBlaster 1.x 22KHz, 8bit, mono. 2. SoundBlaster Pro 1.x 22KHz, 8bit, stereo or 44KHz, 8bit, mono. 3. SoundBlaster 2.0 44KHz, 8bit, mono. 4. SoundBlaster Pro 2.0 stereo 44KHz, 8bit, stereo. 5. not used 6. SoundBlaster 16 44KHz, 16bit, stereo
P	MIDI port. Common values are 300 and 330. Not used by OPENCP.
E	Emu port of AWE32 and compatible. Not used by OPENCP.

Cards like AWE32 or newer are set up like a SB16. Before use, the SoundBlaster cards have to be initialized. These programs need a correctly set BLASTER environment variable.

SB 1.x	
SB 2.0	
SB Pro 1.x	
SB Pro 2.0	
SB 16	To initialize use `sbconfig.exe /s`. You should set the mixer to some useful state with `sb16set.exe /MA:240 /VO:240 /LI:240`. You may also need to set up an env variable like `SET SOUND=C:\SB16` to point to the directory where `sbconfig.exe` and `sb16set.exe` are located.

.2. Gravis Ultra Sound

The Gravis Ultra Sound series need two environment variables.

```
SET ULTRADIR=C:\GUS
SET ULTRASND=220,5,7,7,11
```

`ULTRADIR` points to the directory where the GUS tools are located. The `ULTRASND` variable consists of 5 numbers containing the following information.

1. Base port of GUS. Common values are 220, 240, 260.
2. 1st 16bit DMA channel. Common values are 5, 6, 7.
3. 2nd 16bit DMA channel. Common values are 5, 6, 7.
4. 1st IRQ. Common values are 5, 7, 11.
5. 2nd IRQ. Common values are 5, 7, 11.

GUS initialization is completed by calling `ultrinit.exe`.