# 11.7. Using MEncoder to create QuickTime-compatible files

## 11.7.1. Why would one want to produce QuickTime-compatible Files?

There are several reasons why producing QuickTime-compatible files can be desirable.

- You want any computer illiterate to be able to watch your encode on any major platform (Windows, Mac OS X, Unices …).
- QuickTime is able to take advantage of more hardware and software acceleration features of Mac OS X than platform-independent players like MPlayer or VLC. That means that your encodes have a chance to be played smoothly by older G4-powered machines.
- QuickTime 7 supports the next-generation codec H.264, which yields significantly better picture quality than previous codec generations (MPEG-2, MPEG-4 …).

## 11.7.2. QuickTime 7 limitations

QuickTime 7 supports H.264 video and AAC audio, but it does not support them muxed in the AVI container format. However, you can use MEncoder to encode the video and audio, and then use an external program such as mp4creator (part of the MPEG4IP suite) to remux the video and audio tracks into an MP4 container.

QuickTime's support for H.264 is limited, so you will need to drop some advanced features. If you encode your video with features that QuickTime 7 does not support, QuickTime-based players will show you a pretty white screen instead of your expected video.

- **B-frames**: QuickTime 7 supports a maximum of 1 B-frame, i.e. `-x264encopts bframes=1`. This means that `b_pyramid` and `weight_b` will have no effect, since they require `bframes` to be greater than 1.
- **Macroblocks**: QuickTime 7 does not support 8x8 DCT macroblocks. This option (`8x8dct`) is off by default, so just be sure not to explicitly enable it. This also means that the `i8x8` option will have no effect, since it requires `8x8dct`.
- **Aspect ratio**: QuickTime 7 does not support SAR (sample aspect ratio) information in MPEG-4 files; it assumes that SAR=1. Read the section on scaling for a workaround.

## 11.7.3. Cropping

Suppose you want to rip your freshly bought copy of "The Chronicles of Narnia". Your DVD is region 1, which means it is NTSC. The example below would still apply to PAL, except you would omit `-ofps 24000/1001` and use slightly different `crop` and `scale` dimensions.

After running `mplayer dvd://1`, you follow the process detailed in the section How to deal with telecine and interlacing in NTSC DVDs and discover that it is 24000/1001 fps progressive video. This simplifies the process somewhat, since you do not need to use an inverse telecine filter such as `pullup` or a deinterlacing filter such as `yadif`.

Next, you need to crop out the black bars from the top and bottom of the video, as detailed in this previous section.

## 11.7.4. Scaling

The next step is truly heartbreaking. QuickTime 7 does not support MPEG-4 videos with a sample aspect ratio other than 1, so you will need to upscale (which wastes a lot of disk space) or downscale (which loses some details of the source) the video to square pixels. Either way you do it, this is highly inefficient, but simply cannot be avoided if you want your video to be playable by QuickTime 7. MEncoder can apply the appropriate upscaling or downscaling by specifying respectively `-vf scale=-10:-1` or `-vf scale=-1:-10`. This will scale your video to the correct width for the cropped height, rounded to the closest multiple of 16 for optimal compression. Remember that if you are cropping, you should crop first, then scale:

```
-vf crop=720:352:0:62,scale=-10:-1
```

## 11.7.5. A/V sync

Because you will be remuxing into a different container, you should always use the `harddup` option to ensure that duplicated frames are actually duplicated in the video output. Without this option, MEncoder will simply put a marker in the video stream that a frame was duplicated, and rely on the client software to show the same frame twice. Unfortunately, this "soft duplication" does not survive remuxing, so the audio would slowly lose sync with the video.

The final filter chain looks like this:

```
-vf crop=720:352:0:62,scale=-10:-1,harddup
```

## 11.7.6. Bitrate

As always, the selection of bitrate is a matter of the technical properties of the source, as explained here, as well as a matter of taste. This movie has a fair bit of action and lots of detail, but H.264 video looks good at much lower bitrates than XviD or other MPEG-4 codecs. After much experimentation, the author of this guide chose to encode this movie at 900kbps, and thought that it looked very good. You may decrease bitrate if you need to save more space, or increase it if you need to improve quality.

## 11.7.7. Encoding example

You are now ready to encode the video. Since you care about quality, of course you will be doing a two-pass encode. To shave off some encoding time, you can specify the `turbo` option on the first pass; this reduces `subq` and `frameref` to 1. To save some disk space, you can use the `ss` option to strip off the first few seconds of the video. (I found that this particular movie has 32 seconds of credits and logos.) `bframes` can be 0 or 1. The other options are documented in Encoding with the x264 codec and the man page.

```
mencoder dvd://1 -o /dev/null -ss 32 -ovc x264 \
-x264encopts pass=1:turbo:bitrate=900:bframes=1:\
me=umh:partitions=all:trellis=1:qp_step=4:qcomp=0.7:direct_pred=auto:keyint=300 \
-vf crop=720:352:0:62,scale=-10:-1,harddup \
-oac faac -faacopts br=192:mpeg=4:object=2 -channels 2 -srate 48000 \
-ofps 24000/1001
```

If you have a multi-processor machine, don't miss the opportunity to dramatically speed-up encoding by enabling x264's multi-threading mode by adding `threads=auto` to your `x264encopts` command-line.

The second pass is the same, except that you specify the output file and set `pass=2`.

```
mencoder dvd://1 -o narnia.avi -ss 32 -ovc x264 \
-x264encopts pass=2:turbo:bitrate=900:frameref=5:bframes=1:\
me=umh:partitions=all:trellis=1:qp_step=4:qcomp=0.7:direct_pred=auto:keyint=300 \
-vf crop=720:352:0:62,scale=-10:-1,harddup \
-oac faac -faacopts br=192:mpeg=4:object=2 -channels 2 -srate 48000 \
-ofps 24000/1001
```

The resulting AVI should play perfectly in MPlayer, but of course QuickTime can not play it because it does not support H.264 muxed in AVI. So the next step is to remux the video into an MP4 container.

### 11.7.8. Remuxing as MP4

There are several ways to remux AVI files to MP4. You can use mp4creator, which is part of the MPEG4IP suite.

First, demux the AVI into separate audio and video streams using MPlayer.

```
mplayer narnia.avi -dumpaudio -dumpfile narnia.aac
mplayer narnia.avi -dumpvideo -dumpfile narnia.h264
```

The file names are important; mp4creator requires that AAC audio streams be named `.aac` and H.264 video streams be named `.h264`.

Now use mp4creator to create a new MP4 file out of the audio and video streams.

```
mp4creator -create=narnia.aac narnia.mp4
mp4creator -create=narnia.h264 -rate=23.976 narnia.mp4
```

Unlike the encoding step, you must specify the framerate as a decimal (such as 23.976), not a fraction (such as 24000/1001).

This `narnia.mp4` file should now be playable with any QuickTime 7 application, such as QuickTime Player or iTunes. If you are planning to view the video in a web browser with the QuickTime plugin, you should also hint the movie so that the QuickTime plugin can start playing it while it is still downloading. mp4creator can create these hint tracks:

```
mp4creator -hint=1 narnia.mp4
mp4creator -hint=2 narnia.mp4
mp4creator -optimize narnia.mp4
```

You can check the final result to ensure that the hint tracks were created successfully:

```
mp4creator -list narnia.mp4
```

You should see a list of tracks: 1 audio, 1 video, and 2 hint tracks.

```
Track   Type    Info
1       audio   MPEG-4 AAC LC, 8548.714 secs, 190 kbps, 48000 Hz
2       video   H264 Main@5.1, 8549.132 secs, 899 kbps, 848x352 @ 23.976001 fps
3       hint    Payload mpeg4-generic for track 1
4       hint    Payload H264 for track 2
```

### 11.7.9. Adding metadata tags

If you want to add tags to your video that show up in iTunes, you can use AtomicParsley.

```
AtomicParsley narnia.mp4 --metaEnema --title "The Chronicles of Narnia" --year 2005 --stik Movie --freefree --overWrite
```

The `--metaEnema` option removes any existing metadata (mp4creator inserts its name in the "encoding tool" tag), and `--freefree` reclaims the space from the deleted metadata. The `--stik` option sets the type of video (such as Movie or TV Show), which iTunes uses to group related video files. The `--overWrite` option overwrites the original file; without it, AtomicParsley creates a new auto-named file in the same directory and leaves the original file untouched.

Prev
11.6. Encoding with the `Video For Windows` codec family
Up
Home
Next
11.8. Using MEncoder to create VCD/SVCD/DVD-compliant files