

105d67985f ▾

...

gpac / share / doc / man / gpac.1



jeanlf allow multiple ranges in mp4box split options - cf #2186 ✓

History

2 contributors



4710 lines (4556 sloc) | 95.4 KB

...

```

1  .TH gpac 1 2019 gpac GPAC
2  .
3  .SH NAME
4  .LP
5  gpac \- GPAC command-line filter session manager
6  .SH SYNOPSIS
7  .LP
8  .B gpac
9  .RI [options] FILTER [LINK] FILTER [...]
10 .br
11 .
12 gpac is GPAC's command line tool for setting up and running filter chains.
13 .br
14
15 .br
16 FILTER: a single filter declaration (e.g., -i file, -o dump, inspect, ...), see gpac -h doc.
17 .br
18 [LINK]: a link instruction (e.g., @, @2, @2#StreamType=Visual, ...), see gpac -h doc.
19 .br
20 [options]: one or more option strings, each starting with a - character.
21 .br
22 - an option using a single - indicates an option of gpac (see gpac -hx) or of libgpac (see gpac
23 .br
24 - an option using -- indicates a global filter or meta-filter (e.g. FFmpeg) option, e.g. --block
25 .br
26
27 .br
28 Filter declaration order may impact the link resolver which will try linking in declaration order.
29 .br

```

Options do not require any specific order, and may be present anywhere, including between link sta

.br

Boolean values do not need any value specified. Other types shall be formatted as opt=val, except

.br

.br

The session can be interrupted at any time using ctrl+c, which can also be used to toggle global r

.br

.br

The possible options for gpac are:

.br

.br

.TP

.B \-mem-track

.br

enable memory tracker

.br

.TP

.B \-mem-track-stack

.br

enable memory tracker with stack dumping

.br

.TP

.B \-ltf

.br

load test-unit filters (used for for unit tests only)

.br

.TP

.B \-sloop (int)

.br

loop execution of session, creating a session at each loop, mainly used for testing. If no value i

.br

.TP

.B \-runfor (int)

.br

run for the given amount of milliseconds

.br

.TP

.B \-runforx (int)

.br

run for the given amount of milliseconds and exit with no cleanup

.br

.TP

.B \-runfors (int)

.br

run for the given amount of milliseconds and exit with segfault (tests)

.br

```
79 .TP
80 .B \-runfor1 (int)
81 .br
82 run for the given amount of milliseconds and wait forever at end (tests)
83 .br
84 .TP
85 .B \-stats
86 .br
87 print stats after execution
88 .br
89 .TP
90 .B \-graph
91 .br
92 print graph after execution
93 .br
94 .TP
95 .B \-k
96 .br
97 enable keyboard interaction from command line
98 .br
99 .TP
100 .B \-r (string)
101 .br
102 enable reporting
103 .br
104 * r: runtime reporting
105 .br
106 * r=FA[,FB]: runtime reporting but only print given filters, e.g. r=mp4mx for ISOBMFF multiplexer
107 .br
108 * r=: only print final report
109 .br
110 .TP
111 .B \-seps (string, default: :=#,!@)
112 .br
113 set the default character sets used to separate various arguments
114 .br
115 - the first char is used to separate argument names
116 .br
117 - the second char, if present, is used to separate names and values
118 .br
119 - the third char, if present, is used to separate fragments for PID sources
120 .br
121 - the fourth char, if present, is used for list separators (sourceIDs, gfreq, ...)
122 .br
123 - the fifth char, if present, is used for boolean negation
124 .br
125 - the sixth char, if present, is used for LINK directives (see filters help (-h doc))
126 .br
127 .TP
```

```
128 .B \-i,-src (string)
129 .br
130 specify an input file - see filters help (-h doc)
131 .br
132 .TP
133 .B \-o,-dst (string)
134 .br
135 specify an output file - see filters help (-h doc)
136 .br
137 .TP
138 .B \-ib (string)
139 .br
140 specify an input file to wrap as GF_FileIO object (testing of GF_FileIO)
141 .br
142 .TP
143 .B \-ob (string)
144 .br
145 specify an output file to wrap as GF_FileIO object (testing of GF_FileIO)
146 .br
147 .TP
148 .B \-cl
149 .br
150 force complete mode when no link directive are set - see filters help (-h doc)
151 .br
152 .TP
153 .B \-step
154 .br
155 test step mode in non-blocking session
156 .br
157 .TP
158 .B \-h,-help,-ha,-hx,-hh (string)
159 .br
160 print help. Use -help or -h for basic options, -ha for advanced options, -hx for expert options an
161 .br
162 Note: The @ character can be used in place of the * character. String parameter can be:
163 .br
164 * empty: print command line options help
165 .br
166 * doc: print the general filter info
167 .br
168 * alias: print the gpac alias syntax
169 .br
170 * log: print the log system help
171 .br
172 * core: print the supported libgpac core options. Use -ha/-hx/-hh for advanced/expert options
173 .br
174 * cfg: print the GPAC configuration help
175 .br
176 * prompt: print the GPAC prompt help when running in interactive mode (see .I -k )
```

```
177 .br
178 * modules: print available modules
179 .br
180 * filters: print name of all available filters
181 .br
182 * filters*: print name of all available filters, including meta filters
183 .br
184 * codecs: print the supported builtin codecs - use -hx to include unmapped codecs (ffmpeg, ...)
185 .br
186 * props: print the supported builtin PID and packet properties
187 .br
188 * props PNAME: print the supported builtin PID and packet properties mentioning PNAME
189 .br
190 * colors: print the builtin color names and their values
191 .br
192 * exts: print the builtin extensions and mime types - use -hx to include meta filters (ffmpeg, ...)
193 .br
194 * layouts: print the builtin CICP audio channel layout names and their values
195 .br
196 * links: print possible connections between each supported filters (use -hx to view src->dst cap b
197 .br
198 * links FNAME: print sources and sinks for filter FNAME (either builtin or JS filter)
199 .br
200 * FNAME: print filter FNAME info (multiple FNAME can be given)
201 .br
202 - For meta-filters, use FNAME:INST, e.g. ffavin:avfoundation
203 .br
204 - Use * to print info on all filters (big output!), *: * to print info on all filters including m
205 .br
206 - By default only basic filter options and description are shown. Use -ha to show advanced optio
207 .br
208 * FNAME.OPT: print option OPT in filter FNAME
209 .br
210 * OPT: look in filter names and options for OPT and suggest possible matches if none found. Use -h
211 .br
212
213 .br
214 .TP
215 .B \-p (string)
216 .br
217 use indicated profile for the global GPAC config. If not found, config file is created. If a file
218 .br
219 .TP
220 .B \-alias (string)
221 .br
222 assign a new alias or remove an alias. Can be specified several times. See alias usage (-h alias)
223 .br
224 .TP
225 .B \-aliasdoc (string)
```

```
226 .br
227 assign documentation for a given alias (optional). Can be specified several times
228 .br
229 .TP
230 .B \-uncache
231 .br
232 revert all items in GPAC cache directory to their original name and server path
233 .br
234 .TP
235 .B \-js (string)
236 .br
237 specify javascript file to use as controller of filter session
238 .br
239 .TP
240 .B \-wc
241 .br
242 write all core options in the config file unless already set
243 .br
244 .TP
245 .B \-we
246 .br
247 write all file extensions in the config file unless already set (useful to change some default fil
248 .br
249 .TP
250 .B \-wf
251 .br
252 write all filter options in the config file unless already set
253 .br
254 .TP
255 .B \-wfx
256 .br
257 write all filter options and all meta filter arguments in the config file unless already set (larg
258 .br
259 .TP
260 .B \-xopt
261 .br
262 unrecognized options and filters declaration following this option are ignored - used to pass argu
263 .br
264
265 .br
266
267 .br
268 The following libgpac core options allow customizing the filter session:
269 .br
270
271 .br
272 .TP
273 .B \-dbg-edges
274 .br
```

```
275 log edges status in filter graph before dijkstra resolution (for debug). Edges are logged as edge_
276 .br
277 .TP
278 .B \-full-link
279 .br
280 throw error if any PID in the filter graph cannot be linked
281 .br
282 .TP
283 .B \-no-block (Enum, default: no)
284 .br
285 disable blocking mode of filters
286 .br
287 * no: enable blocking mode
288 .br
289 * fanout: disable blocking on fan-out, unblocking the PID as soon as one of its destinations requi
290 .br
291 * all: disable blocking
292 .br
293 .TP
294 .B \-no-reg
295 .br
296 disable regulation (no sleep) in session
297 .br
298 .TP
299 .B \-no-reassign
300 .br
301 disable source filter reassignment in PID graph resolution
302 .br
303 .TP
304 .B \-sched (Enum, default: free)
305 .br
306 set scheduler mode
307 .br
308 * free: lock-free queues except for task list (default)
309 .br
310 * lock: mutexes for queues when several threads
311 .br
312 * freex: lock-free queues including for task lists (experimental)
313 .br
314 * flock: mutexes for queues even when no thread (debug mode)
315 .br
316 * direct: no threads and direct dispatch of tasks whenever possible (debug mode)
317 .br
318 .TP
319 .B \-max-chain (int, default: 6)
320 .br
321 set maximum chain length when resolving filter links. Default value covers for [ in -> ] dmx -> re
322 .br
323 .TP
```

```
324 .B \-max-sleep (int, default: 50)
325 .br
326 set maximum sleep time slot in milliseconds when regulation is enabled
327 .br
328 .TP
329 .B \-threads (int)
330 .br
331 set N extra thread for the session. -1 means use all available cores
332 .br
333 .TP
334 .B \-no-probe
335 .br
336 disable data probing on sources and relies on extension (faster load but more error-prone)
337 .br
338 .TP
339 .B \-no-argchk
340 .br
341 disable tracking of argument usage (all arguments will be considered as used)
342 .br
343 .TP
344 .B \-blacklist (string)
345 .br
346 blacklist the filters listed in the given string (comma-separated list). If first character is '-'
347 .br
348 .TP
349 .B \-no-graph-cache
350 .br
351 disable internal caching of filter graph connections. If disabled, the graph will be recomputed at
352 .br
353 .TP
354 .B \-no-reservoir
355 .br
356 disable memory recycling for packets and properties. This uses much less memory but stresses the s
357 .br
358 .SH Using Aliases
359 .PL
360 The gpac command line can become quite complex when many sources or filters are used. In order to
361 .br
362 .br
363 .br
364 To assign an alias, use the syntax gpac -alias="NAME VALUE".
365 .br
366 * `NAME`: shall be a single string, with no space.
367 .br
368 * `VALUE`: the list of argument this alias replaces. If not set, the alias is destroyed
369 .br
370 .br
371 .br
372 When parsing arguments, the alias will be replace by its value.
```



```
373 .br
374 Example
375 .br
376 gpac -alias="output aout vout"
377 .br
378
379 .br
380 This allows later audio and video playback using gpac -i src.mp4 output
381 .br
382
383 .br
384 Aliases can use arguments from the command line. The allowed syntaxes are:
385 .br
386 * @{a}`: replaced by the value of the argument with index a after the alias
387 .br
388 * @{a,b}`: replaced by the value of the arguments with index a and b
389 .br
390 * @{a:b}`: replaced by the value of the arguments between index a and b
391 .br
392 * @{-a,b}`: replaced by the value of the arguments with index a and b, inserting a list separator
393 .br
394 * @{-a:b}`: replaced by the value of the arguments between index a and b, inserting a list separator
395 .br
396 * @{+a,b}`: clones the parent word in the alias for a and b, replacing this pattern in each clone
397 .br
398 * @{+a:b}`: clones the parent word in the alias for each argument between index a and b, replacing
399 .br
400
401 .br
402 The specified index can be:
403 .br
404 * forward index: a strictly positive integer, 1 being the first argument after the alias
405 .br
406 * backward index: the value 'n' (or 'N') to indicate the last argument on the command line. This c
407 .br
408
409 .br
410 Before solving aliases, all option arguments are moved at the beginning of the command line. This
411 .br
412 Arguments not used by any aliases are kept on the command line, other ones are removed
413 .br
414
415 .br
416 Example
417 .br
418 -alias="foo src=@{N} dst=test.mp4"
419 .br
420
421 .br
```

```
422 The command gpac foo f1 f2 expands to gpac src=f2 dst=test.mp4 f1
423 .br
424 Example
425 .br
426 -alias="list: inspect src=@{+:N}"
427 .br
428
429 .br
430 The command gpac list f1 f2 f3 expands to gpac inspect src=f1 src=f2 src=f3
431 .br
432 Example
433 .br
434 -alias="list inspect src=@{+2:N}"
435 .br
436
437 .br
438 The command gpac list f1 f2 f3 expands to gpac inspect src=f2 src=f3 f1
439 .br
440 Example
441 .br
442 -alias="plist aout vout flist:srcs=@{-,N}"
443 .br
444
445 .br
446 The command gpac plist f1 f2 f3 expands to gpac aout vout flist:srcs="f1,f2,f3"
447 .br
448
449 .br
450 Alias documentation can be set using gpac -aliasdoc="NAME VALUE", with NAME the alias name and VAL
451 .br
452 Alias documentation will then appear in gpac help.
453 .br
454
455 .br
456 .SH Configuration file
457 .LP
458 .br
459 GPAC uses a configuration file to modify default options of libgpac and filters. This file is call
460 .br
461 - on Windows platforms, in C:\Users\F00\AppData\Roaming\GPAC or in C:\Program Files\GPAC.
462 .br
463 - on iOS platforms, in a .gpac folder in the app storage directory.
464 .br
465 - on Android platforms, in /sdcard/GPAC/ if this directory exists, otherwise in /data/data/io.gpac
466 .br
467 - on other platforms, in a $HOME/.gpac/.
468 .br
469
470 .br
```

471 Applications in GPAC can also specify a different configuration file through the `.I -p profile opt`
472 `.br`
473 This will load configuration from `$HOME/.gpac/foo/GPAC.cfg`, creating it if needed.
474 `.br`
475 The reserved name `0` is used to disable configuration file writing.
476 `.br`
477
478 `.br`
479 The configuration file is structured in sections, each made of one or more keys:
480 `.br`
481 - section `foo` is declared as `[foo]\n`
482 `.br`
483 - key `bar` with value `N` is declared as `bar=N\n`. The key value `N` is not interpreted and always handl
484 `.br`
485
486 `.br`
487 By default the configuration file only holds a few system specific options and directories. It is
488 `.br`
489 This should be avoided as the resulting configuration file size will be quite large, hence larger
490 `.br`
491 The options specified in the configuration file may be overridden by the values in `restrict.cfg` fi
492 `.br`
493 Note: The methods describe in this section apply to any application in GPAC transferring their arg
494 `.br`
495
496 `.br`
497 **.SH Core options**
498 `.LP`
499 `.br`
500 The options from `libgpac` core can also be assigned though the config file from section `core` using
501 `.br`
502 Example
503 `.br`
504 `[core]`
505 `.br`
506 `threads=2`
507 `.br`
508
509 `.br`
510 Setting this in the config file is equivalent to using `-threads=2`.
511 `.br`
512 The options specified at prompt overrides the value of the config file.
513 `.br`
514 **.SH Filter options in configuration**
515 `.LP`
516 `.br`
517 It is possible to alter the default value of a filter option by modifying the configuration file.
518 `.br`
519 Example

```
520 .br
521 [filter@rtppin]
522 .br
523 interleave=yes
524 .br
525
526 .br
527 This will force the rtp input filter to always request RTP over RTSP by default.
528 .br
529 To generate a configuration file with all filters options serialized, use .I -wf.
530 .br
531 .SH Global filter options
532 .LP
533 .br
534 It is possible to specify options global to multiple filters using --OPTNAME=VAL. Global options d
535 .br
536 This will set option OPTNAME, when present, to VAL in any loaded filter.
537 .br
538 Example
539 .br
540 --buffer=100 -i file vout aout
541 .br
542
543 .br
544 This is equivalent to specifying vout:buffer=100 aout:buffer=100.
545 .br
546 Example
547 .br
548 --buffer=100 -i file vout aout:buffer=10
549 .br
550
551 .br
552 This is equivalent to specifying vout:buffer=100 aout:buffer=10.
553 .br
554 Warning: This syntax only applies to regular filter options. It cannot be used with builtin shortc
555 .br
556 Meta-filter options can be set in the same way using the syntax --OPT_NAME=VAL.
557 .br
558 Example
559 .br
560 --profile=Baseline -i file.cmp -o dump.264
561 .br
562
563 .br
564 This is equivalent to specifying -o dump.264:profile=Baseline.
565 .br
566
567 .br
568 For both syntaxes, it is possible to specify the filter registry name of the option, using --FNAME
```

```
569 .br
570 In this case the option will only be set for filters which are instances of registry FNAME. This i
571 .br
572 Example
573 .br
574 --flist@timescale=100 -i plist1 -i plist2 -o live.mpd
575 .br
576
577 .br
578 This will set the timescale option on the playlists filters but not on the dasher filter.
579 .br
580 .SH libgpac core options:
581 .LP
582 .br
583 .TP
584 .B \-noprog
585 .br
586 disable progress messages
587 .br
588 .TP
589 .B \-quiet
590 .br
591 disable all messages, including errors
592 .br
593 .TP
594 .B \-proglf
595 .br
596 use new line at each progress messages
597 .br
598 .TP
599 .B \-strict-error,-se
600 .br
601 exit after the first error is reported
602 .br
603 .TP
604 .B \-store-dir (string)
605 .br
606 set storage directory
607 .br
608 .TP
609 .B \-mod-dirs (string list)
610 .br
611 set additional module directories as a semi-colon ; separated list
612 .br
613 .TP
614 .B \-js-dirs (string list)
615 .br
616 set javascript directories
617 .br
```

```
618 .TP
619 .B \-no-js-mods (string)
620 .br
621 disable javascript module loading
622 .br
623 .TP
624 .B \-ifce (string)
625 .br
626 set default multicast interface through interface IP address (default is 127.0.0.1)
627 .br
628 .TP
629 .B \-lang (string)
630 .br
631 set preferred language
632 .br
633 .TP
634 .B \-cfg,-opt (string)
635 .br
636 get or set configuration file value. The string parameter can be formatted as:
637 .br
638 * `section:key=val`: set the key to a new value
639 .br
640 * `section:key=null`, `section:key`: remove the key
641 .br
642 * `section=null`: remove the section
643 .br
644 * no argument: print the entire configuration file
645 .br
646 * `section`: print the given section
647 .br
648 * `section:key`: print the given key in section (section can be set to *)- *:key: print the given
649 .br
650 .TP
651 .B \-no-save
652 .br
653 discard any changes made to the config file upon exit
654 .br
655 .TP
656 .B \-mod-reload
657 .br
658 unload / reload module shared libs when no longer used
659 .br
660 .TP
661 .B \-for-test
662 .br
663 disable all creation/modification dates and GPAC versions in files
664 .br
665 .TP
666 .B \-old-arch
```

```
667 .br
668 enable compatibility with pre-filters versions of GPAC
669 .br
670 .TP
671 .B \-ntp-shift (int)
672 .br
673 shift NTP clock by given amount in seconds
674 .br
675 .TP
676 .B \-bs-cache-size (int, default: 512)
677 .br
678 cache size for bitstream read and write from file (0 disable cache, slower IOs)
679 .br
680 .TP
681 .B \-no-check
682 .br
683 disable compliance tests for inputs (ISOBMFF for now). This will likely result in random crashes
684 .br
685 .TP
686 .B \-unhandled-rejection
687 .br
688 dump unhandled promise rejections
689 .br
690 .TP
691 .B \-startup-file (string)
692 .br
693 startup file of compositor in GUI mode
694 .br
695 .TP
696 .B \-docs-dir (string)
697 .br
698 default documents directoty (for GUI on iOS and Android)
699 .br
700 .TP
701 .B \-last-dir (string)
702 .br
703 last working directory (for GUI)
704 .br
705 .TP
706 .B \-cache (string)
707 .br
708 cache directory location
709 .br
710 .TP
711 .B \-proxy-on
712 .br
713 enable HTTP proxy
714 .br
715 .TP
```

```
716 .B \-proxy-name (string)
717 .br
718 set HTTP proxy address
719 .br
720 .TP
721 .B \-proxy-port (int, default: 80)
722 .br
723 set HTTP proxy port
724 .br
725 .TP
726 .B \-maxrate (int)
727 .br
728 set max HTTP download rate in bits per sec. 0 means unlimited
729 .br
730 .TP
731 .B \-no-cache
732 .br
733 disable HTTP caching
734 .br
735 .TP
736 .B \-offline-cache
737 .br
738 enable offline HTTP caching (no re-validation of existing resource in cache)
739 .br
740 .TP
741 .B \-clean-cache
742 .br
743 indicate if HTTP cache should be clean upon launch/exit
744 .br
745 .TP
746 .B \-cache-size (int, default: 100M)
747 .br
748 specify cache size in bytes
749 .br
750 .TP
751 .B \-head-timeout (int, default: 5000)
752 .br
753 set HTTP head request timeout in milliseconds
754 .br
755 .TP
756 .B \-req-timeout (int, default: 20000)
757 .br
758 set HTTP/RTSP request timeout in milliseconds
759 .br
760 .TP
761 .B \-no-timeout
762 .br
763 ignore HTTP 1.1 timeout in keep-alive
764 .br
```



```
765 .TP
766 .B \-broken-cert
767 .br
768 enable accepting broken SSL certificates
769 .br
770 .TP
771 .B \-user-agent,-ua (string)
772 .br
773 set user agent name for HTTP/RTSP
774 .br
775 .TP
776 .B \-user-profileid (string)
777 .br
778 set user profile ID (through X-UserProfileID entity header) in HTTP requests
779 .br
780 .TP
781 .B \-user-profile (string)
782 .br
783 set user profile filename. Content of file is appended as body to HTTP HEAD/GET requests, associat
784 .br
785 .TP
786 .B \-query-string (string)
787 .br
788 insert query string (without ?) to URL on requests
789 .br
790 .TP
791 .B \-dm-threads
792 .br
793 force using threads for async download requests rather than session scheduler
794 .br
795 .TP
796 .B \-cte-rate-wnd (int, default: 20)
797 .br
798 set window analysis length in milliseconds for chunk-transfer encoding rate estimation
799 .br
800 .TP
801 .B \-no-h2
802 .br
803 disable HTTP2
804 .br
805 .TP
806 .B \-no-h2c
807 .br
808 disable HTTP2 upgrade (i.e. over non-TLS)
809 .br
810 .TP
811 .B \-h2-copy
812 .br
813 enable intermediate copy of data in nghttp2 (default is disabled but may report as broken frames i
```

```
814 .br
815 .TP
816 .B \-dbg-edges
817 .br
818 log edges status in filter graph before dijkstra resolution (for debug). Edges are logged as edge_
819 .br
820 .TP
821 .B \-full-link
822 .br
823 throw error if any PID in the filter graph cannot be linked
824 .br
825 .TP
826 .B \-no-block (Enum, default: no)
827 .br
828 disable blocking mode of filters
829 .br
830 * no: enable blocking mode
831 .br
832 * fanout: disable blocking on fan-out, unblocking the PID as soon as one of its destinations requi
833 .br
834 * all: disable blocking
835 .br
836 .TP
837 .B \-no-reg
838 .br
839 disable regulation (no sleep) in session
840 .br
841 .TP
842 .B \-no-reassign
843 .br
844 disable source filter reassignment in PID graph resolution
845 .br
846 .TP
847 .B \-sched (Enum, default: free)
848 .br
849 set scheduler mode
850 .br
851 * free: lock-free queues except for task list (default)
852 .br
853 * lock: mutexes for queues when several threads
854 .br
855 * freex: lock-free queues including for task lists (experimental)
856 .br
857 * flock: mutexes for queues even when no thread (debug mode)
858 .br
859 * direct: no threads and direct dispatch of tasks whenever possible (debug mode)
860 .br
861 .TP
862 .B \-max-chain (int, default: 6)
```

```
863 .br
864 set maximum chain length when resolving filter links. Default value covers for [ in -> ] dmx -> re
865 .br
866 .TP
867 .B \-max-sleep (int, default: 50)
868 .br
869 set maximum sleep time slot in milliseconds when regulation is enabled
870 .br
871 .TP
872 .B \-threads (int)
873 .br
874 set N extra thread for the session. -1 means use all available cores
875 .br
876 .TP
877 .B \-no-probe
878 .br
879 disable data probing on sources and relies on extension (faster load but more error-prone)
880 .br
881 .TP
882 .B \-no-argchk
883 .br
884 disable tracking of argument usage (all arguments will be considered as used)
885 .br
886 .TP
887 .B \-blacklist (string)
888 .br
889 blacklist the filters listed in the given string (comma-separated list). If first character is '-'
890 .br
891 .TP
892 .B \-no-graph-cache
893 .br
894 disable internal caching of filter graph connections. If disabled, the graph will be recomputed at
895 .br
896 .TP
897 .B \-no-reservoir
898 .br
899 disable memory recycling for packets and properties. This uses much less memory but stresses the s
900 .br
901 .TP
902 .B \-switch-vres
903 .br
904 select smallest video resolution larger than scene size, otherwise use current video resolution
905 .br
906 .TP
907 .B \-hwvmem (Enum, default: auto)
908 .br
909 specify (2D rendering only) memory type of main video backbuffer. Depending on the scene type, thi
910 .br
911 * always: always on hardware
```

```
912 .br
913 * never: always on system memory
914 .br
915 * auto: selected by GPAC based on content type (graphics or video)
916 .br
917 .TP
918 .B \-pref-yuv4cc (string)
919 .br
920 set preferred YUV 4CC for overlays (used by DirectX only)
921 .br
922 .TP
923 .B \-offscreen-yuv
924 .br
925 indicate if offscreen yuv->rgb is enabled. can be set to false to force disabling
926 .br
927 .TP
928 .B \-overlay-color-key (string)
929 .br
930 color to use for overlay keying, hex format
931 .br
932 .TP
933 .B \-gl-bits-comp (int, default: 8)
934 .br
935 number of bits per color component in OpenGL
936 .br
937 .TP
938 .B \-gl-bits-depth (int, default: 16)
939 .br
940 number of bits for depth buffer in OpenGL
941 .br
942 .TP
943 .B \-gl-doublebuf
944 .br
945 enable OpenGL double buffering
946 .br
947 .TP
948 .B \-sdl-defer
949 .br
950 use defer rendering for SDL
951 .br
952 .TP
953 .B \-no-colorkey
954 .br
955 disable color keying at the video output level
956 .br
957 .TP
958 .B \-glfbo-txid (int)
959 .br
960 set output texture ID when using glfbo output. The OpenGL context shall be initialized and gf_term
```

961 .br
962 .TP
963 **.B \-video-output (string)**
964 .br
965 indicate the name of the video output module to use (see `gpac -h modules`). The reserved name `glfbo`
966 .br
967 .TP
968 **.B \-dfb-sys (string, default: x11)**
969 .br
970 system DirectFB (`x11`, `sdl`, `vnc`, `fbdev`, `osx` or `devmem`)
971 .br
972 .TP
973 **.B \-dfb-flip (string, default: waitsync)**
974 .br
975 vsync mode for DirectFB (`waitsync`, `wait`, `sync` or `swap`)
976 .br
977 .TP
978 **.B \-audio-output (string)**
979 .br
980 indicate the name of the audio output module to use
981 .br
982 .TP
983 **.B \-alsa-devname (string)**
984 .br
985 set ALSA dev name
986 .br
987 .TP
988 **.B \-force-alsarate (int)**
989 .br
990 force ALSA and OSS output sample rate
991 .br
992 .TP
993 **.B \-ds-disable-notif**
994 .br
995 disable DirectSound audio buffer notifications when supported
996 .br
997 .TP
998 **.B \-font-reader (string)**
999 .br
1000 indicate name of font reader module
1001 .br
1002 .TP
1003 **.B \-font-dirs (string)**
1004 .br
1005 indicate comma-separated list of directories to scan for fonts
1006 .br
1007 .TP
1008 **.B \-rescan-fonts**
1009 .br

```
1010 indicate the font directory must be rescanned
1011 .br
1012 .TP
1013 .B \-wait-fonts
1014 .br
1015 wait for SVG fonts to be loaded before displaying frames
1016 .br
1017 .TP
1018 .B \-webvtt-hours
1019 .br
1020 force writing hour when serializing WebVTT
1021 .br
1022 .TP
1023 .B \-charset (string)
1024 .br
1025 set charset when not recognized from input. Possible values are:
1026 .br
1027 * utf8: force UTF-8
1028 .br
1029 * utf16: force UTF-16 little endian
1030 .br
1031 * utf16be: force UTF-16 big endian
1032 .br
1033 * other: attempt to parse anyway
1034 .br
1035 .TP
1036 .B \-rmt
1037 .br
1038 enable profiling through Remotery. A copy of Remotery visualizer is in gpac/share/vis, usually ins
1039 .br
1040 .TP
1041 .B \-rmt-port (int, default: 17815)
1042 .br
1043 set remotery port
1044 .br
1045 .TP
1046 .B \-rmt-reuse
1047 .br
1048 allow remotery to reuse port
1049 .br
1050 .TP
1051 .B \-rmt-localhost
1052 .br
1053 make remotery only accepts localhost connection
1054 .br
1055 .TP
1056 .B \-rmt-sleep (int, default: 10)
1057 .br
1058 set remotery sleep (ms) between server updates
```

```
1059 .br
1060 .TP
1061 .B \-rmt-nmsg (int, default: 10)
1062 .br
1063 set remotery number of messages per update
1064 .br
1065 .TP
1066 .B \-rmt-qsize (int, default: 131072)
1067 .br
1068 set remotery message queue size in bytes
1069 .br
1070 .TP
1071 .B \-rmt-log
1072 .br
1073 redirect logs to remotery (experimental, usually not well handled by browser)
1074 .br
1075 .TP
1076 .B \-rmt-ogl
1077 .br
1078 make remotery sample opengl calls
1079 .br
1080 .TP
1081 .B \-m2ts-vvc-old
1082 .br
1083 hack for old TS streams using 0x32 for VVC instead of 0x33
1084 .br
1085 .TP
1086 .B \-piff-force-subsamples
1087 .br
1088 hack for PIFF PSEC files generated by 0.9.0 and 1.0 MP4Box with wrong subsample_count inserted for
1089 .br
1090 .TP
1091 .B \-vvdec-annexb
1092 .br
1093 hack for old vvdec+libavcodec supporting only annexB format
1094 .br
1095 .SH libgpac logs options:
1096 .LP
1097 .br
1098 .TP
1099 .B \-noprog
1100 .br
1101 disable progress messages
1102 .br
1103 .TP
1104 .B \-quiet
1105 .br
1106 disable all messages, including errors
1107 .br
```

```
1108 .TP
1109 .B \-log-file,-lf (string)
1110 .br
1111 set output log file
1112 .br
1113 .TP
1114 .B \-log-clock,-lc
1115 .br
1116 log time in micro sec since start time of GPAC before each log line except for app tool
1117 .br
1118 .TP
1119 .B \-log-utc,-lu
1120 .br
1121 log UTC time in ms before each log line except for app tool
1122 .br
1123 .TP
1124 .B \-logs (string)
1125 .br
1126 set log tools and levels.
1127 .br
1128
1129 .br
1130 You can independently log different tools involved in a session.
1131 .br
1132 log_args is formatted as a colon (':') separated list of toolX[:toolZ]@levelX
1133 .br
1134 levelX can be one of:
1135 .br
1136 * quiet: skip logs
1137 .br
1138 * error: logs only error messages
1139 .br
1140 * warning: logs error+warning messages
1141 .br
1142 * info: logs error+warning+info messages
1143 .br
1144 * debug: logs all messages
1145 .br
1146
1147 .br
1148 toolX can be one of:
1149 .br
1150 * core: libgpac core
1151 .br
1152 * mutex: log all mutex calls
1153 .br
1154 * mem: GPAC memory tracker
1155 .br
1156 * module: GPAC modules (av out, font engine, 2D rasterizer)
```



```
1157 .br
1158 * filter: filter session debugging
1159 .br
1160 * sched: filter session scheduler debugging
1161 .br
1162 * codec: codec messages (used by encoder and decoder filters)
1163 .br
1164 * coding: bitstream formats (audio, video, scene)
1165 .br
1166 * container: container formats (ISO File, MPEG-2 TS, AVI, ...) and multiplexer/demultiplexer filters
1167 .br
1168 * network: TCP/UDP sockets and TLS
1169 .br
1170 * http: HTTP traffic
1171 .br
1172 * cache: HTTP cache subsystem
1173 .br
1174 * rtp: RTP traffic
1175 .br
1176 * dash: HTTP streaming logs
1177 .br
1178 * route: ROUTE (ATSC3) debugging
1179 .br
1180 * media: messages from generic filters and reframer/rewriter filters
1181 .br
1182 * parser: textual parsers (svg, xmt, bt, ...)
1183 .br
1184 * mmio: I/O management (AV devices, file, pipes, OpenGL)
1185 .br
1186 * audio: audio renderer/mixer/output
1187 .br
1188 * script: script engine except console log
1189 .br
1190 * console: script console log
1191 .br
1192 * scene: scene graph and scene manager
1193 .br
1194 * compose: composition engine (2D, 3D, etc)
1195 .br
1196 * ctime: media and SMIL timing info from composition engine
1197 .br
1198 * interact: interaction messages (UI events and triggered DOM events and VRML route)
1199 .br
1200 * rti: run-time stats of compositor
1201 .br
1202 * all: all tools logged - other tools can be specified afterwards.
1203 .br
1204 The special keyword ncl can be set to disable color logs.
1205 .br
```

```
1206 The special keyword strict can be set to exit at first error.
1207 .br
1208
1209 .br
1210 Example
1211 .br
1212 -logs=all@info:dash@debug:nc1
1213 .br
1214
1215 .br
1216 This moves all log to info level, dash to debug level and disable color logs
1217 .br
1218 .TP
1219 .B \-proglf
1220 .br
1221 use new line at each progress messages
1222 .br
1223 .SH General
1224 .LP
1225 .br
1226 Filters are configurable processing units consuming and producing data packets. These packets are
1227 .br
1228 Note: When a PID cannot be connected to any filter, a warning is thrown and all packets dispatched
1229 .br
1230
1231 .br
1232 Each output PID carries a set of properties describing the data it delivers (e.g. width, height, c
1233 .br
1234
1235 .br
1236 Each filter exposes a set of argument to configure itself, using property types and values describ
1237 .br
1238 .SH Property and filter option format
1239 .LP
1240 .br
1241 * boolean: formatted as yes,true,1 or no,false,0
1242 .br
1243 * enumeration (for filter arguments only): must use the syntax given in the argument description,
1244 .br
1245 * 1-dimension (numbers, floats, ints...): formatted as value[unit], where unit can be k,K (x 1000)
1246 .br
1247 * fraction: formatted as num/den or num-den or num, in which case the denominator is 1 if num is a
1248 .br
1249 * unsigned 32 bit integer: formatted as number or hexadecimal using the format 0xAABCCDD.
1250 .br
1251 * N-dimension (vectors): formatted as DIM1xDIM2[xDIM3[xDIM4]] values, without unit multiplier.
1252 .br
1253 * string: formatted as:
1254 .br
```

```
1255     * `value`: copies value to string.
1256 .br
1257     * `file@FILE`: load string from local FILE (opened in binary mode).
1258 .br
1259     * `bxml@FILE`: binarize XML from local FILE and set property type to data - see https://wiki.gpac
1260 .br
1261     * data: formatted as:
1262 .br
1263     * `size@address`: constant data block, not internally copied; size gives the size of the block,
1264 .br
1265     * `0xBYTESTRING`: data block specified in hexadecimal, internally copied.
1266 .br
1267     * `file@FILE`: load data from local FILE (opened in binary mode).
1268 .br
1269     * `bxml@FILE`: binarize XML from local FILE - see https://wiki.gpac.io/NHML-Format.
1270 .br
1271     * `b64@DATA`: load data from base-64 encoded DATA.
1272 .br
1273     * pointer: pointer address as formatted by %p in C.
1274 .br
1275     * string lists: formatted as val1,val2[,...]. Each value can also use file@FILE syntax.
1276 .br
1277     * integer lists: formatted as val1,val2[,...]
1278 .br
1279     Note: The special characters in property formats (0x,/,-,+I,-I,x) cannot be configured.
1280 .br
1281 .SH Filter declaration [FILTER]
1282 .LP
1283 .br
1284 .SS Generic declaration
1285 .br
1286     Each filter is declared by its name, with optional filter arguments appended as a list of colon-se
1287 .br
1288     * boolean: value can be omitted, defaulting to true (e.g. :noedit). Using ! before the name negate
1289 .br
1290     * enumerations: name can be omitted, e.g. :disp=pbo is equivalent to :pbo.
1291 .br
1292 .br
1293 .br
1294 .br
1295 .br
1296     When string parameters are used (e.g. URLs), it is recommended to escape the string using the keyw
1297 .br
1298     Example
1299 .br
1300     filter:ARG=http://foo/bar?yes:gpac:opt=VAL
1301 .br
1302 .br
1303 .br
```

1304 This will properly extract the URL.
1305 .br
1306 Example
1307 .br
1308 filter:ARG=http://foo/bar?yes:opt=VAL
1309 .br
1310
1311 .br
1312 This will fail to extract it and keep :opt=VAL as part of the URL.
1313 .br
1314 The escape mechanism is not needed for local source, for which file existence is probed during arg
1315 .br
1316 For tcp:// and udp:// protocols, the escape is not needed if a trailing / is appended after the po
1317 .br
1318 Example
1319 .br
1320 -i tcp://127.0.0.1:1234:OPT
1321 .br
1322
1323 .br
1324 This will fail to extract the URL and options.
1325 .br
1326 Example
1327 .br
1328 -i tcp://127.0.0.1:1234/:OPT
1329 .br
1330
1331 .br
1332 This will extract the URL and options.
1333 .br
1334 Note: one trick to avoid the escape sequence is to declare the URLs option at the end, e.g. f1:opt
1335 .br
1336
1337 .br
1338 It is possible to disable option parsing (for string options) by duplicating the separator.
1339 .br
1340 Example
1341 .br
1342 filter::opt1=UDP://IP:PORT/:someopt=VAL::opt2=VAL2
1343 .br
1344
1345 .br
1346 This will pass UDP://IP:PORT/:someopt=VAL to opt1 without inspecting it, and VAL2 to opt2.
1347 .br
1348
1349 .br
1350 **.SS Source and Sink filters**
1351 .br
1352 Source and sink filters do not need to be addressed by the filter name, specifying src= or dst= in

1353 .br
1354 Example
1355 .br
1356 "src=file.mp4" or "-src file.mp4" or "-i file.mp4"
1357 .br
1358
1359 .br
1360 This will find a filter (for example fin) able to load file.mp4. The same result can be achieved by
1361 .br
1362 Example
1363 .br
1364 "dst=dump.yuv" or "-dst dump.yuv" or "-o dump.yuv"
1365 .br
1366
1367 .br
1368 This will dump the video content in dump.yuv. The same result can be achieved by using fout:dst=du
1369 .br
1370
1371 .br
1372 Specific source or sink filters may also be specified using filterName:src=URL or filterName:dst=U
1373 .br
1374
1375 .br
1376 The src= and dst= syntaxes can also be used in alias for dynamic argument cloning (see gpac -hx al
1377 .br
1378
1379 .br
1380 **.SS Forcing specific filters**
1381 .br
1382 There is a special option called gfreg which allows specifying preferred filters to use when handl
1383 .br
1384 Example
1385 .br
1386 src=file.mp4:gfreg=ffdmx,ffdec
1387 .br
1388
1389 .br
1390 This will use fdmx to read file.mp4 and ffdec to decode it.
1391 .br
1392 This can be used to test a specific filter when alternate filter chains are possible.
1393 .br
1394 **.SS Specifying encoders and decoders**
1395 .br
1396 By default filters chain will be resolved without any decoding/encoding if the destination accepts
1397 .br
1398 * c=NAME: identifies the desired codec. NAME can be the GPAC codec name or the encoder instance fo
1399 .br
1400 * b=UINT, rate=UINT, bitrate=UINT: indicates the bitrate in bits per second
1401 .br

1402 * g=UINT, gop=UINT: indicates the GOP size in frames
1403 .br
1404 * pfmt=NAME: indicates the target pixel format name (see properties (-h props)) of the source, if
1405 .br
1406 * all_intra=BOOL: indicates all frames should be intra frames, if supported by codec
1407 .br
1408
1409 .br
1410 Other options will be passed to the filter if it accepts generic argument parsing (as is the case
1411 .br
1412 The shortcut syntax c=TYPE (e.g. c=aac:opts) is also supported.
1413 .br
1414
1415 .br
1416 Example
1417 .br
1418 gpac -i dump.yuv:size=320x240:fps=25 enc:c=avc:b=150000:g=50:cgop=true:fast=true -o raw.264
1419 .br
1420
1421 .br
1422 This creates a 25 fps AVC at 175kbps with a gop duration of 2 seconds, using closed gop and fast e
1423 .br
1424
1425 .br
1426 The inverse operation (forcing a decode to happen) is possible using the reframer filter.
1427 .br
1428 Example
1429 .br
1430 gpac -i file.mp4 reframer:raw=av -o null
1431 .br
1432
1433 .br
1434 This will force decoding media from file.mp4 and trash (send to null) the result (doing a decoder
1435 .br
1436
1437 .br
1438 **.SS Escaping option separators**
1439 .br
1440 When a filter uses an option defined as a string using the same separator character as gpac, you c
1441 .br
1442 Example
1443 .br
1444 f:a=foo:b=bar
1445 .br
1446
1447 .br
1448 This will set option a to foo and option b to bar on the filter.
1449 .br
1450 Example

```
1451 .br
1452 f::a=foo:b=bar
1453 .br
1454
1455 .br
1456 This will set option a to foo:b=bar on the filter.
1457 .br
1458 Example
1459 .br
1460 f:a=foo::b=bar:c::d=fun
1461 .br
1462
1463 .br
1464 This will set option a to foo, b to bar:c and the option d to fun on the filter.
1465 .br
1466
1467 .br
1468 .SH Filter linking [LINK]
1469 .LP
1470 .br
1471
1472 .br
1473 Each filter exposes one or more sets of capabilities, called capability bundle, which are property
1474 .br
1475 To check the possible sources and destination for a filter FNAME, use gpac -h links FNAME
1476 .br
1477
1478 .br
1479 The filter graph resolver uses this information together with the PID properties to link the diffe
1480 .br
1481
1482 .br
1483 Link directives, when provided, specify which source a filter can accept connections from.
1484 .br
1485 They do not specify which destination a filter can connect to.
1486 .br
1487
1488 .br
1489 .SS Default filter linking
1490 .br
1491 When no link instructions are given (see below), the default linking strategy used is either impli
1492 .br
1493 Each PID is checked for possible connection to all defined filters, in their declaration order.
1494 .br
1495 For each filter DST accepting a connection from the PID, directly or with intermediate filters:
1496 .br
1497 - if DST filter has link directives, use them to allow or reject PID connection.
1498 .br
1499 - otherwise, if complete mode is enabled, allow connection.
```

```
1500 .br
1501 - otherwise (implicit mode):
1502 .br
1503 - if DST is not a sink and is the first matching filter with no link directive, allow connection.
1504 .br
1505 - otherwise, if DST is not a sink and is not the first matching filter with no link directive, re
1506 .br
1507 - otherwise (DST is a sink) and no previous connections to a non-sink filter, allow connection.
1508 .br
1509
1510 .br
1511 Example
1512 .br
1513 gpac -i file.mp4 c=avc -o output
1514 .br
1515
1516 .br
1517 With this setup in implicit mode:
1518 .br
1519 - if the file has a video PID, it will connect to enc but not to output. The output PID of enc wil
1520 .br
1521 - if the file has other PIDs than video, they will connect to output, since this enc filter accept
1522 .br
1523
1524 .br
1525 Example
1526 .br
1527 gpac -cl -i file.mp4 c=avc -o output
1528 .br
1529
1530 .br
1531 With this setup in complete mode:
1532 .br
1533 - if the file has a video PID, it will connect both to enc and to output, and the output PID of en
1534 .br
1535 - if the file has other PIDs than video, they will connect to output.
1536 .br
1537
1538 .br
1539 Furthermore in implicit mode, filter connections are restricted to filters defined between the las
1540 .br
1541 Example
1542 .br
1543 gpac -i video1 reframer:saps=1 -i video2 ffsws:osize=128x72 -o output
1544 .br
1545
1546 .br
1547 This will connect:
1548 .br
```


1549 - video1 to reframer then reframer to output but will prevent reframer to ffsws connection.
1550 .br
1551 - video2 to ffsws then ffsws to output but will prevent video2 to reframer connection.
1552 .br
1553
1554 .br
1555 Example
1556 .br
1557 gpac -i video1 -i video2 reframer:saps=1 ffsws:osize=128x72 -o output
1558 .br
1559
1560 .br
1561 This will connect video1 AND video2 to reframer->ffsws->output
1562 .br
1563
1564 .br
1565 The implicit mode allows specifying linear processing chains (no PID fan-out except for final outp
1566 .br
1567 Warning: Argument order really matters in implicit mode!
1568 .br
1569
1570 .br
1571 Example
1572 .br
1573 gpac -i file.mp4 c=avc c=aac -o output
1574 .br
1575
1576 .br
1577 If the file has a video PID, it will connect to c=avc but not to output. The output PID of c=avc w
1578 .br
1579 If the file has an audio PID, it will connect to c=aac but not to output. The output PID of c=aac v
1580 .br
1581 If the file has other PIDs than audio or video, they will connect to output.
1582 .br
1583
1584 .br
1585 Example
1586 .br
1587 gpac -i file.mp4 ffswf=osize:128x72 c=avc resample=osr=48k c=aac -o output
1588 .br
1589
1590 .br
1591 This will force:
1592 .br
1593 - SRC(video)->ffsws->enc(video)->output and prevent SRC(video)->output, SRC(video)->enc(video) and
1594 .br
1595 - SRC(audio)->resample->enc(audio)->output and prevent SRC(audio)->output, SRC(audio)->enc(audio)
1596 .br
1597

1598 .br

1599 **.SS Quick links**

1600 .br

1601 Link between filters may be manually specified. The syntax is an @ character optionally followed by

1602 .br

1603 This indicates that the following filter specified at prompt should be linked only to a previous 1

1604 .br

1605 The optional integer is a 0-based index to the previous filter declarations, 0 indicating the prev

1606 .br

1607 If @@ is used instead of @, the optional integer gives the filter index starting from the first fi

1608 .br

1609 Several link directives can be given for a filter.

1610 .br

1611 Example

1612 .br

1613 fA fB @1 fC

1614 .br

1615

1616 .br

1617 This indicates that fC only accepts inputs from fA.

1618 .br

1619 Example

1620 .br

1621 fA fB fC @1 @@ fD

1622 .br

1623

1624 .br

1625 This indicates that fD only accepts inputs from fB and fC.

1626 .br

1627 Example

1628 .br

1629 fA fB fC ... @@1 fZ

1630 .br

1631

1632 .br

1633 This indicates that fZ only accepts inputs from fB.

1634 .br

1635

1636 .br

1637 **.SS Complex links**

1638 .br

1639 The @ link directive is just a quick shortcut to set the following filter arguments:

1640 .br

1641 * FID=name: assigns an identifier to the filter

1642 .br

1643 * SID=name1[,name2...]: sets a list of filter identifiers, or sourceIDs, restricting the list of p

1644 .br

1645

1646 .br

1647 Example
1648 .br
1649 fA fB @1 fC
1650 .br
1651
1652 .br
1653 This is equivalent to fA:FID=1 fB fC:SID=1.
1654 .br
1655 Example
1656 .br
1657 fA:FID=1 fB fC:SID=1
1658 .br
1659
1660 .br
1661 This indicates that fC only accepts input from fA, but fB might accept inputs from fA.
1662 .br
1663 Example
1664 .br
1665 fA:FID=1 fB:FID=2 fC:SID=1 fD:SID=1,2
1666 .br
1667
1668 .br
1669 This indicates that fD only accepts input from fA and fB and fC only from fA
1670 .br
1671 Note: A filter with sourceID set cannot get input from filters with no IDs.
1672 .br
1673
1674 .br
1675 A sourceID name can be further extended using fragment identifier (# by default):
1676 .br
1677 * name#PIDNAME: accepts only PID(s) with name PIDNAME
1678 .br
1679 * name#TYPE: accepts only PIDs of matching media type. TYPE can be audio, video, scene, text, font
1680 .br
1681 * name#TYPEN: accepts only N (1-based index) PID of matching type from source (e.g. video2 to only
1682 .br
1683 * name#TAG=VAL: accepts the PID if its parent filter has no tag or a tag matching VAL
1684 .br
1685 * name#P4CC=VAL: accepts only PIDs with builtin property of type P4CC and value VAL.
1686 .br
1687 * name#PName=VAL: same as above, using the builtin name corresponding to the property.
1688 .br
1689 * name#AnyName=VAL: same as above, using the name of a non built-in property.
1690 .br
1691 * name#Name=OtherPropName: compares the value with the value of another property of the PID. The m
1692 .br
1693 If the property is not defined on the PID, the property is matched. Otherwise, its value is checke
1694 .br
1695

1696 .br
1697 The following modifiers for comparisons are allowed (for any fragment format using =):
1698 .br
1699 * name#P4CC=!VAL: accepts only PIDs with property NOT matching VAL.
1700 .br
1701 * name#P4CC-VAL: accepts only PIDs with property strictly less than VAL (only for 1-dimension numb
1702 .br
1703 * name#P4CC+VAL: accepts only PIDs with property strictly greater than VAL (only for 1-dimension n
1704 .br
1705
1706 .br
1707 A sourceID name can also use wildcard or be empty to match a property regardless of the source fil
1708 .br
1709 Example
1710 .br
1711 fA fB:SID=*#ServiceID=2
1712 .br
1713 fA fB:SID=#ServiceID=2
1714 .br
1715
1716 .br
1717 This indicates to match connection between fA and fB only for PIDs with a ServiceID property of 2.
1718 .br
1719 These extensions also work with the LINK @ shortcut.
1720 .br
1721 Example
1722 .br
1723 fA fB @1#video fC
1724 .br
1725
1726 .br
1727 This indicates that fC only accepts inputs from fA, and of type video.
1728 .br
1729 Example
1730 .br
1731 gpac -i img.heif @#ItemID=200 vout
1732 .br
1733
1734 .br
1735 This indicates to connect to vout only PIDs with ItemID property equal to 200.
1736 .br
1737 Example
1738 .br
1739 gpac -i vid.mp4 @#PID=1 vout
1740 .br
1741
1742 .br
1743 This indicates to connect to vout only PIDs with ID property equal to 1.
1744 .br

1745 Example
1746 .br
1747 gpac -i vid.mp4 @#Width=640 vout
1748 .br
1749
1750 .br
1751 This indicates to connect to vout only PIDs with Width property equal to 640.
1752 .br
1753 Example
1754 .br
1755 gpac -i vid.mp4 @#Width-640 vout
1756 .br
1757
1758 .br
1759 This indicates to connect to vout only PIDs with Width property less than 640
1760 .br
1761 Example
1762 .br
1763 gpac -i vid.mp4 @#ID=ItemID#ItemNumber=1 vout
1764 .br
1765
1766 .br
1767 This will connect to vout only PID with an ID property equal to ItemID property (keep items, disca
1768 .br
1769
1770 .br
1771 Multiple fragment can be specified to check for multiple PID properties.
1772 .br
1773 Example
1774 .br
1775 gpac -i vid.mp4 @#Width=640#Height+380 vout
1776 .br
1777
1778 .br
1779 This indicates to connect to vout only PIDs with Width property equal to 640 and Height greater th
1780 .br
1781
1782 .br
1783 Warning: If a PID directly connects to one or more explicitly loaded filters, no further dynamic l
1784 .br
1785 Example
1786 .br
1787 fA @ reframer fB
1788 .br
1789
1790 .br
1791 If fB accepts inputs provided by fA but reframer does not, this will link fA PID to fB filter sinc
1792 .br
1793 Since the PID is connected, the filter engine will not try to solve a link between fA and reframer

1794 .br
1795
1796 .br
1797 An exception is made for local files: by default, a local file destination will force a remultiplex.
1798 .br
1799 Example
1800 .br
1801 `gpac -i file.mp4 -o dump.mp4`
1802 .br
1803
1804 .br
1805 This will prevent direct connection of PID of type file to dst file.mp4, remultiplexing the file.
1806 .br
1807
1808 .br
1809 The special option nomux is used to allow direct connections (ignored for non-sink filters).
1810 .br
1811 Example
1812 .br
1813 `gpac -i file.mp4 -o dump.mp4:nomux`
1814 .br
1815
1816 .br
1817 This will result in a direct file copy.
1818 .br
1819
1820 .br
1821 This only applies to local files destination. For pipes, sockets or other file outputs (HTTP, ROUTE),
1822 .br
1823 - direct copy is enabled by default
1824 .br
1825 - nomux=0 can be used to force remultiplex
1826 .br
1827
1828 .br
1829 **.SS Sub-session tagging**
1830 .br
1831 Filters may be assigned to a sub-session using :FS=N, with N a positive integer.
1832 .br
1833 Filters belonging to different sub-sessions may only link to each-other:
1834 .br
1835 - if explicitly allowed through sourceID directives (@ or SID)
1836 .br
1837 - or if they have the same sub-session identifier
1838 .br
1839
1840 .br
1841 This is mostly used for implicit mode in gpac: each first source filter specified after a sink filter
1842 .br

1843 Example
1844 .br
1845 gpac -i in1.mp4 -i in2.mp4 -o out1.mp4 -o out2.mp4
1846 .br
1847
1848 .br
1849 This will result in both inputs multiplexed in both outputs.
1850 .br
1851 Example
1852 .br
1853 gpac -i in1.mp4 -o out1.mp4 -i in2.mp4 -o out2.mp4
1854 .br
1855
1856 .br
1857 This will result in in1 mixed to out1 and in2 mixed to out2, these last two filters belonging to a
1858 .br
1859
1860 .br
1861 **.SH Arguments inheriting**
1862 .LP
1863 .br
1864 Unless explicitly disabled (see .I -max-chain), the filter engine will resolve implicit or explicit
1865 .br
1866 Example
1867 .br
1868 gpac -i file.mp4:OPT -o file.aac -o file.264
1869 .br
1870
1871 .br
1872 This will pass the :OPT to all filters loaded between the source and the two destinations.
1873 .br
1874 Example
1875 .br
1876 gpac -i file.mp4 -o file.aac:OPT -o file.264
1877 .br
1878
1879 .br
1880 This will pass the :OPT to all filters loaded between the source and the file.aac destination.
1881 .br
1882 Note: the destination arguments inherited are the arguments placed AFTER the dst= option.
1883 .br
1884 Example
1885 .br
1886 gpac -i file.mp4 fout:OPTFOO:dst=file.aac:OPTBAR
1887 .br
1888
1889 .br
1890 This will pass the :OPTBAR to all filters loaded between file.mp4 source and file.aac destination,
1891 .br

1892 Arguments inheriting can be stopped by using the keyword gfloc: arguments after the keyword will n
1893 .br
1894 Example
1895 .br
1896 gpac -i file.mp4 -o file.aac:OPTF00:gfloc:OPTBAR -o file.264
1897 .br
1898
1899 .br
1900 This will pass :OPTF00 to all filters loaded between file.mp4 source and file.aac destination, but
1901 .br
1902 Arguments are by default tracked to check if they were used by the filter chain, and a warning is
1903 .br
1904 It may be useful to specify arguments which may not be consumed depending on the graph resolution;
1905 .br
1906 Example
1907 .br
1908 gpac -i file.mp4 -o file.aac:OPTF00:gfopt:OPTBAR -o file.264
1909 .br
1910
1911 .br
1912 This will warn if OPTF00 is not consumed, but will not track OPTBAR.
1913 .br
1914
1915 .br
1916 A filter may be assigned a name (for inspection purposes, not inherited) using :N=name option. Thi
1917 .br
1918
1919 .br
1920 A filter may be assigned a tag (any string) using :TAG=name option. This tag does not need to be u
1921 .br
1922
1923 .br
1924 .SH URL templating
1925 .LP
1926 .br
1927 Destination URLs can be dynamically constructed using templates. Pattern \$KEYWORD\$ is replaced in
1928 .br
1929 KEYWORD is case sensitive, and may be present multiple times in the string. Supported KEYWORD:
1930 .br
1931 * num: replaced by file number if defined, 0 otherwise
1932 .br
1933 * PID: ID of the source PID
1934 .br
1935 * URL: URL of source file
1936 .br
1937 * File: path on disk for source file; if not found, use URL if set, or PID name otherwise
1938 .br
1939 * Type: name of stream type of PID (video, audio ...)
1940 .br

1941 * p4cc=ABCD: uses PID property with 4CC value ABCD
1942 .br
1943 * pname=VAL: uses PID property with name VAL
1944 .br
1945 * OTHER: locates property 4CC for the given name, or property name if no 4CC matches.
1946 .br
1947
1948 .br
1949 \$\$ is an escape for \$
1950 .br
1951
1952 .br
1953 Templating can be useful when encoding several qualities in one pass.
1954 .br
1955 Example
1956 .br
1957 gpac -i dump.yuv:size=640x360 vcrop:wnd=0x0x320x180 c=avc:b=1M @2 c=avc:b=750k -o dump_\$CropOrigin
1958 .br
1959
1960 .br
1961 This will create a cropped version of the source, encoded in AVC at 1M, and a full version of the
1962 .br
1963 **.SH Cloning filters**
1964 .LP
1965 .br
1966 When a filter accepts a single connection and has a connected input, it is no longer available for
1967 .br
1968 Example
1969 .br
1970 gpac -i img.heif -o dump_\$ItemID\$.jpg
1971 .br
1972
1973 .br
1974 In this case, only one item (likely the first declared in the file) will connect to the destination
1975 .br
1976 Other items will not be connected since the destination only accepts one input PID.
1977 .br
1978 There is a special option clone allowing filters to be cloned with the same arguments. The cloned
1979 .br
1980 Example
1981 .br
1982 gpac -i img.heif -o dump_\$ItemID\$.jpg:clone
1983 .br
1984
1985 .br
1986 In this case, the destination will be cloned for each item, and all will be exported to different
1987 .br
1988 Example
1989 .br

1990 gpac -i vid.mpd c=avc:FID=1:clone -o transcode.mpd:SID=1
1991 .br
1992
1993 .br
1994 In this case, the encoder will be cloned for each video PIDs in the source, and the destination wi
1995 .br
1996
1997 .br
1998 When implicit linking is enabled, all filters are by default clonable. This allows duplicating the
1999 .br
2000 Example
2001 .br
2002 gpac -i dual_audio resample:osr=48k c=aac -o dst
2003 .br
2004
2005 .br
2006 The resampler filter will be cloned for each audio PID, and the encoder will be cloned for each re
2007 .br
2008 You can explicitly deactivate the cloning instructions:
2009 .br
2010 Example
2011 .br
2012 gpac -i dual_audio resample:osr=48k:clone=0 c=aac -o dst
2013 .br
2014
2015 .br
2016 The first audio will connect to the resample filter, the second to the enc filter and the resample
2017 .br
2018
2019 .br
2020 **.SH Templating filter chains**
2021 .LP
2022 .br
2023 There can be cases where the number of desired outputs depends on the source content, for example
2024 .br
2025 To handle this, it is possible to use a PID property name in the sourceID of a filter with the val
2026 .br
2027 Warning: This feature should only be called with a single property set to * (or empty) per source
2028 .br
2029 Example
2030 .br
2031 gpac -i source.ts -o file_\${ServiceID\$.mp4:SID=##ServiceID=*
2032 .br
2033 gpac -i source.ts -o file_\${ServiceID\$.mp4:SID=#ServiceID=
2034 .br
2035
2036 .br
2037 In this case, each new ServiceID value found when connecting PIDs to the destination will create a
2038 .br

2039
2040 .br
2041 Cloning in implicit linking mode applies to output as well:
2042 .br
2043 Example
2044 .br
2045 gpac -i dual_audio -o dst_\${PID\$.aac
2046 .br
2047
2048 .br
2049 Each audio track will be dumped to aac (potentially reencoding if needed).
2050 .br
2051
2052 .br
2053 **.SH Assigning PID properties**
2054 .LP
2055 .br
2056 It is possible to define properties on output PIDs that will be declared by a filter. This allows
2057 .br
2058 This sets output PIDs property (4cc, built-in name or any name) to the given value. Value can be o
2059 .br
2060 Non built-in properties are parsed as follows:
2061 .br
2062 - file@F00 will be declared as string with a value set to the content of F00.
2063 .br
2064 - bxml@F00 will be declared as data with a value set to the binarized content of F00.
2065 .br
2066 - F00 will be declared as string with a value set to F00.
2067 .br
2068 - TYPE@F00 will be parsed according to TYPE. If the type is not recognized, the entire value is co
2069 .br
2070
2071 .br
2072 User-assigned PID properties on filter fA will be inherited by all filters dynamically loaded to s
2073 .br
2074 If fB also has user-assigned PID properties, these only apply starting from fB in the chain and ar
2075 .br
2076
2077 .br
2078 Warning: Properties are not filtered and override the properties of the filter's output PIDs, be c
2079 .br
2080 Example
2081 .br
2082 gpac -i v1.mp4:#ServiceID=4 -i v2.mp4:#ServiceID=2 -o dump.ts
2083 .br
2084
2085 .br
2086 This will multiplex the streams in dump.ts, using ServiceID 4 for PIDs from v1.mp4 and ServiceID 2
2087 .br

2088
2089 .br
2090 PID properties may be conditionally assigned by checking other PID properties. The syntax uses par
2091 .br
2092 #Prop=(CP=CV)VAL
2093 .br
2094 This will assign PID property Prop to VAL for PIDs with property CP equal to CV.
2095 .br
2096 #Prop=(CP=CV)VAL,(CP2=CV2)VAL2
2097 .br
2098 This will assign PID property Prop to VAL for PIDs with property CP equal to CV, and to VAL2 for P
2099 .br
2100 #Prop=(CP=CV)(CP2=CV2)VAL
2101 .br
2102 This will assign PID property Prop to VAL for PIDs with property CP equal to CV and property CP2 e
2103 .br
2104 #Prop=(CP=CV)VAL,()DEFAULT
2105 .br
2106 This will assign PID property Prop to VAL for PIDs with property CP equal to CV, or to DEFAULT for
2107 .br
2108 The condition syntax is the same as source ID fragment syntax.
2109 .br
2110 Note: When set, the default value (empty condition) always matches the PID, therefore it should be
2111 .br
2112 Example
2113 .br
2114 gpac -i source.mp4:#MyProp=(audio)"Super Audio",(video)"Super Video"
2115 .br
2116
2117 .br
2118 This will assign property MyProp to Super Audio for audio PIDs and to Super Video for video PIDs.
2119 .br
2120 Example
2121 .br
2122 gpac -i source.mp4:#MyProp=(audio1)"Super Audio"
2123 .br
2124
2125 .br
2126 This will assign property MyProp to Super Audio for first audio PID declared.
2127 .br
2128 Example
2129 .br
2130 gpac -i source.mp4:#MyProp=(Width+1280)HD
2131 .br
2132
2133 .br
2134 This will assign property MyProp to HD for PIDs with property Width greater than 1280.
2135 .br
2136 .SH Using option files

2137 .LP
2138 .br
2139 It is possible to use a file to define options of a filter, by specifying the target file name as
2140 .br
2141 Warning: Only local files are allowed.
2142 .br
2143 An option file is a simple text file containing one or more options or PID properties on one or mo
2144 .br
2145 A line beginning with "/" is a comment and is ignored.
2146 .br
2147 Options in an option file may point to other option files, with a maximum redirection level of 5.
2148 .br
2149 An option file declaration (filter:myopts.txt) follows the same inheritance rules as regular optio
2150 .br
2151 Example
2152 .br
2153 gpac -i source.mp4:myopts.txt:foo=bar -o dst
2154 .br
2155
2156 .br
2157 Any filter loaded between source.mp4 and dst will inherit both myopts.txt and foo options and will
2158 .br
2159 **.SH Specific filter options**
2160 .LP
2161 .br
2162 Some specific keywords are replaced when processing filter options.
2163 .br
2164 Warning: These keywords do not apply to PID properties. Multiple keywords cannot be defined for a
2165 .br
2166 Defined keywords:
2167 .br
2168 * \$GSHARE: replaced by system path to GPAC shared directory (e.g. /usr/share/gpac)
2169 .br
2170 * \$GJS: replaced by the first path from global share directory and paths set through .I -js-dirs t
2171 .br
2172 * \$GDOCS: replaced by system path to:
2173 .br
2174 - application document directory for iOS
2175 .br
2176 - EXTERNAL_STORAGE environment variable if present or /sdcard otherwise for Android
2177 .br
2178 - user home directory for other platforms
2179 .br
2180 * \$GLANG: replaced by the global config language option .I -lang
2181 .br
2182 * \$GUA: replaced by the global config user agent option .I -user-agent
2183 .br
2184 * \$GINC(init_val[,inc]): replaced by init_val and increment init_val by inc (positive or negative
2185 .br

2186
2187 .br
2188 The \$GINC construct can be used to dynamically assign numbers in filter chains:
2189 .br
2190 Example
2191 .br
2192 gpac -i source.ts tssplit @#ServiceID= -o dump_\$GINC(10,2).ts
2193 .br
2194
2195 .br
2196 This will dump first service in dump_10.ts, second service in dump_12.ts, etc...
2197 .br
2198
2199 .br
2200 As seen previously, the following options may be set on any filter, but are not visible in individ
2201 .br
2202 * FID: filter identifier
2203 .br
2204 * SID: filter source(s)
2205 .br
2206 * N: filter name
2207 .br
2208 * FS: sub-session identifier
2209 .br
2210 * TAG: filter tag
2211 .br
2212 * clone: filter cloning flag
2213 .br
2214 * nomux: enable/disable direct file copy
2215 .br
2216 * gfreg: preferred filter registry names for link solving
2217 .br
2218 * gfloc: following options are local to filter declaration (not inherited)
2219 .br
2220 * gfopt: following options are not tracked
2221 .br
2222 * gpac: argument separator for URLs
2223 .br
2224 ...
2225 .br
2226 .SH External filters
2227 .LP
2228 .br
2229 GPAC comes with a set of built-in filters in libgpac. It may also load external filters in dynamic
2230 .br
2231
2232 .br
2233 .SH GPAC Built-in properties
2234 .LP

2235 .br
2236 Built-in property types
2237 .br
2238 .TP
2239 **.B sint**
2240 .br
2241 signed 32 bit integer
2242 .br
2243 .TP
2244 **.B uint**
2245 .br
2246 unsigned 32 bit integer
2247 .br
2248 .TP
2249 **.B lsint**
2250 .br
2251 signed 64 bit integer
2252 .br
2253 .TP
2254 **.B luint**
2255 .br
2256 unsigned 32 bit integer
2257 .br
2258 .TP
2259 **.B bool**
2260 .br
2261 boolean
2262 .br
2263 .TP
2264 **.B frac**
2265 .br
2266 32/32 bit fraction
2267 .br
2268 .TP
2269 **.B lfrac**
2270 .br
2271 64/64 bit fraction
2272 .br
2273 .TP
2274 **.B flt**
2275 .br
2276 32 bit float number
2277 .br
2278 .TP
2279 **.B dbl**
2280 .br
2281 64 bit float number
2282 .br
2283 .TP

2284	.B v2di
2285	.br
2286	2D 32-bit integer vector
2287	.br
2288	.TP
2289	.B v2d
2290	.br
2291	2D 64-bit float vector
2292	.br
2293	.TP
2294	.B v3di
2295	.br
2296	3D 32-bit integer vector
2297	.br
2298	.TP
2299	.B v4di
2300	.br
2301	4D 32-bit integer vector
2302	.br
2303	.TP
2304	.B str
2305	.br
2306	UTF-8 string
2307	.br
2308	.TP
2309	.B mem
2310	.br
2311	data buffer
2312	.br
2313	.TP
2314	.B cstr
2315	.br
2316	const UTF-8 string
2317	.br
2318	.TP
2319	.B cmem
2320	.br
2321	const data buffer
2322	.br
2323	.TP
2324	.B ptr
2325	.br
2326	32 or 64 bit pointer
2327	.br
2328	.TP
2329	.B strl
2330	.br
2331	UTF-8 string list
2332	.br


```
2333 .TP
2334 .B uint1
2335 .br
2336 unsigned 32 bit integer list
2337 .br
2338 .TP
2339 .B sint1
2340 .br
2341 signed 32 bit integer list
2342 .br
2343 .TP
2344 .B v2il
2345 .br
2346 2D 32-bit integer vector list
2347 .br
2348 .TP
2349 .B 4cc
2350 .br
2351 Four character code
2352 .br
2353 .TP
2354 .B 4cc1
2355 .br
2356 four-character codes list
2357 .br
2358 .TP
2359 .B pfmt
2360 .br
2361 raw pixel format
2362 .br
2363 .TP
2364 .B afmt
2365 .br
2366 raw audio format
2367 .br
2368 .TP
2369 .B cprm
2370 .br
2371 color primaries, string or int value from ISO/IEC 23091-2
2372 .br
2373 .TP
2374 .B ctfc
2375 .br
2376 color transfer characteristics, string or int value from ISO/IEC 23091-2
2377 .br
2378 .TP
2379 .B cmxc
2380 .br
2381 color matrix coefficients, string or int value from ISO/IEC 23091-2
```

```

2382 .br
2383
2384 .br
2385 Built-in properties for PIDs and packets listed as Name (4CC type FLAGS): description
2386 .br
2387 FLAGS can be D (droppable - see GSF multiplexer filter help), P (packet property)
2388 .br
2389 .TP
2390 .B ID (PIDI,uint, )
2391 .br
2392 Stream ID
2393 .br
2394 .TP
2395 .B ESID (ESID,uint,D )
2396 .br
2397 MPEG-4 ESID of PID
2398 .br
2399 .TP
2400 .B ItemID (ITID,uint, )
2401 .br
2402 ID of image item in HEIF, same value as ID
2403 .br
2404 .TP
2405 .B ItemNumber (ITIX,uint, )
2406 .br
2407 Number (1-based) of image item in HEIF, in order of declaration in file
2408 .br
2409 .TP
2410 .B TrackNumber (PIDX,uint, )
2411 .br
2412 Number (1-based) of track in order of declaration in file
2413 .br
2414 .TP
2415 .B ServiceID (PSID,uint,D )
2416 .br
2417 ID of parent service
2418 .br
2419 .TP
2420 .B ClockID (CKID,uint,D )
2421 .br
2422 ID of clock reference PID
2423 .br
2424 .TP
2425 .B DependencyID (DPID,uint, )
2426 .br
2427 ID of layer depended on
2428 .br
2429 .TP
2430 .B SubLayer (DPSL,bool, )

```

```
2431 .br
2432 PID is a sublayer of the stream depended on rather than an enhancement layer
2433 .br
2434 .TP
2435 .B PlaybackMode (PBKM,uint,D )
2436 .br
2437 Playback mode supported:
2438 .br
2439 * 0: no time control
2440 .br
2441 * 1: play/pause/seek,speed=1
2442 .br
2443 * 2: play/pause/seek,speed>=0
2444 .br
2445 * 3: play/pause/seek, reverse playback
2446 .br
2447 .TP
2448 .B Scalable (SCAL,bool, )
2449 .br
2450 Scalable stream
2451 .br
2452 .TP
2453 .B TileBase (SABT,bool, )
2454 .br
2455 Tile base stream
2456 .br
2457 .TP
2458 .B TileID (PTID,uint, )
2459 .br
2460 ID of the tile for hvt1/hvt2 PIDs
2461 .br
2462 .TP
2463 .B Language (LANG,cstr, )
2464 .br
2465 Language code: ISO639 2/3 character code or RFC 4646
2466 .br
2467 .TP
2468 .B ServiceName (SNAM,str,D )
2469 .br
2470 Name of parent service
2471 .br
2472 .TP
2473 .B ServiceProvider (SPRO,str,D )
2474 .br
2475 Provider of parent service
2476 .br
2477 .TP
2478 .B StreamType (PMST,uint, )
2479 .br
```

2480 Media stream type
2481 .br
2482 .TP
2483 **.B StreamSubtype (PSST,4cc,D)**
2484 .br
2485 Media subtype 4CC (auxiliary, pic sequence, etc ..), matches ISOM handler type
2486 .br
2487 .TP
2488 **.B ISOMSubtype (PIST,4cc,D)**
2489 .br
2490 ISOM media subtype 4CC (avc1 avc2...)
2491 .br
2492 .TP
2493 **.B OrigStreamType (POST,uint,)**
2494 .br
2495 Original stream type before encryption
2496 .br
2497 .TP
2498 **.B CodecID (POTI,uint,)**
2499 .br
2500 Codec ID (MPEG-4 OTI or ISOBMFF 4CC)
2501 .br
2502 .TP
2503 **.B InitialObjectDescriptor (PIOD,bool,)**
2504 .br
2505 PID is declared in the IOD for MPEG-4
2506 .br
2507 .TP
2508 **.B Unframed (PFRM,bool,)**
2509 .br
2510 The media data is not framed, i.e. each packet is not a complete AU/frame or is not in internal fo
2511 .br
2512 .TP
2513 **.B UnframedAU (PFRF,bool,)**
2514 .br
2515 The unframed media still has correct AU boundaries: one packet is one full AU, but the packet form
2516 .br
2517 .TP
2518 **.B LATM (LATM,bool,)**
2519 .br
2520 Media is unframed AAC in LATM format
2521 .br
2522 .TP
2523 **.B Duration (PDUR,lfrac,)**
2524 .br
2525 Media duration (a negative value means an estimated duration based on rate)
2526 .br
2527 .TP
2528 **.B NumFrames (NFRM,uint,D)**

```
2529 .br
2530 Number of frames in the stream
2531 .br
2532 .TP
2533 .B FrameOffset (FRMO,uint,D )
2534 .br
2535 Index of first frame in the stream (used for reporting)
2536 .br
2537 .TP
2538 .B ConstantFrameSize (CFRS,uint, )
2539 .br
2540 Size of the frames for constant frame size streams
2541 .br
2542 .TP
2543 .B TimeshiftDepth (PTSD,frac,D )
2544 .br
2545 Depth of the timeshift buffer
2546 .br
2547 .TP
2548 .B TimeshiftTime (PTST,dbl,D )
2549 .br
2550 Time in the timeshift buffer in seconds - changes are signaled through PID info (no reconfigure)
2551 .br
2552 .TP
2553 .B TimeshiftState (PTSS,uint,D )
2554 .br
2555 State of timeshift buffer: 0 is OK, 1 is underflow, 2 is overflow - changes are signaled through P
2556 .br
2557 .TP
2558 .B Timescale (TIMS,uint, )
2559 .br
2560 Media timescale (a timestamp delta of N is N/timescale seconds)
2561 .br
2562 .TP
2563 .B ProfileLevel (PRPL,uint,D )
2564 .br
2565 MPEG-4 profile and level
2566 .br
2567 .TP
2568 .B DecoderConfig (DCFG,mem, )
2569 .br
2570 Decoder configuration data
2571 .br
2572 .TP
2573 .B DecoderConfigEnhancement (ECFG,mem, )
2574 .br
2575 Decoder configuration data of the enhancement layer(s). Also used by 3GPP/Apple text streams to gi
2576 .br
2577 .TP
```

2578 **.B DecoderConfigIndex** (ICFG,uint,)
2579 .br
2580 1-based index of decoder config for ISO base media files
2581 .br
2582 .TP
2583 **.B SampleRate** (AUSR,uint,)
2584 .br
2585 Audio sample rate
2586 .br
2587 .TP
2588 **.B SamplesPerFrame** (FRMS,uint,)
2589 .br
2590 Number of audio sample in one coded frame
2591 .br
2592 .TP
2593 **.B NumChannels** (CHNB,uint,)
2594 .br
2595 Number of audio channels
2596 .br
2597 .TP
2598 **.B BPS** (ABPS,uint,)
2599 .br
2600 Number of bits per sample in compressed source
2601 .br
2602 .TP
2603 **.B ChannelLayout** (CHLO,luint,)
2604 .br
2605 Channel Layout mask
2606 .br
2607 .TP
2608 **.B AudioFormat** (AFMT,afmt,)
2609 .br
2610 Audio sample format
2611 .br
2612 .TP
2613 **.B AudioPlaybackSpeed** (ASPD,dbl,D)
2614 .br
2615 Audio playback speed, only used for audio output reconfiguration
2616 .br
2617 .TP
2618 **.B Delay** (MDLY,lsint,)
2619 .br
2620 Delay of presentation compared to composition timestamps, in media timescale. Positive value imply
2621 .br
2622 .TP
2623 **.B CTSShift** (MDTS,uint,)
2624 .br
2625 CTS offset to apply in case of negative ctts
2626 .br

2627 .TP
2628 .B SkipPriming (ASKP,bool,)
2629 .br
2630 Audio priming shall not to be removed when initializing decoding
2631 .br
2632 .TP
2633 .B Width (WIDT,uint,)
2634 .br
2635 Visual Width (video / text / graphics)
2636 .br
2637 .TP
2638 .B Height (HEIG,uint,)
2639 .br
2640 Visual Height (video / text / graphics)
2641 .br
2642 .TP
2643 .B PixelFormat (PFMT,pfmt,)
2644 .br
2645 Pixel format
2646 .br
2647 .TP
2648 .B PixelFormatWrapped (PFMW,pfmt,)
2649 .br
2650 Underlying pixel format of video stream if pixel format is external GL texture
2651 .br
2652 .TP
2653 .B Stride (VSTY,uint,)
2654 .br
2655 Image or Y/alpha plane stride
2656 .br
2657 .TP
2658 .B StrideUV (VSTC,uint,)
2659 .br
2660 UV plane or U/V planes stride
2661 .br
2662 .TP
2663 .B BitDepthLuma (YBPS,uint,)
2664 .br
2665 Bit depth for luma components
2666 .br
2667 .TP
2668 .B BitDepthChroma (CBPS,uint,)
2669 .br
2670 Bit depth for chroma components
2671 .br
2672 .TP
2673 .B FPS (VFPPF,frac,)
2674 .br
2675 Video framerate

```
2676 .br
2677 .TP
2678 .B Interlaced (VILC,bool, )
2679 .br
2680 Video is interlaced
2681 .br
2682 .TP
2683 .B SAR (PSAR,frac, )
2684 .br
2685 Sample (i.e. pixel) aspect ratio
2686 .br
2687 .TP
2688 .B PAR (VPAR,frac,D )
2689 .br
2690 Picture aspect ratio
2691 .br
2692 .TP
2693 .B MaxWidth (MWID,uint, )
2694 .br
2695 Maximum width (video / text / graphics) of all enhancement layers
2696 .br
2697 .TP
2698 .B MaxHeight (MHEI,uint, )
2699 .br
2700 Maximum height (video / text / graphics) of all enhancement layers
2701 .br
2702 .TP
2703 .B ZOrder (VZIX,sint, )
2704 .br
2705 Z-order of the video, from 0 (first) to max int (last)
2706 .br
2707 .TP
2708 .B TransX (VTRX,sint, )
2709 .br
2710 Horizontal translation of the video (positive towards right)
2711 .br
2712 .TP
2713 .B TransY (VTRY,sint, )
2714 .br
2715 Vertical translation of the video (positive towards up)
2716 .br
2717 .TP
2718 .B TransXRight (VTRx,sint, )
2719 .br
2720 Horizontal offset of the video from right (positive towards right), for cases where reference width
2721 .br
2722 .TP
2723 .B TransYTop (VTRy,sint, )
2724 .br
```



```

2725 Vertical translation of the video (0 is top, positive towards down), for cases where reference hei
2726 .br
2727 .TP
2728 .B Hidden (HIDE,bool, )
2729 .br
2730 PID is hidden in visual/audio rendering
2731 .br
2732 .TP
2733 .B CropOrigin (VCXY,v2di, )
2734 .br
2735 Position in source window, X,Y indicates coord in source
2736 .br
2737 .TP
2738 .B OriginalSize (VOWH,v2di, )
2739 .br
2740 Original resolution of video
2741 .br
2742 .TP
2743 .B SRD (SRD ,v4di, )
2744 .br
2745 Position and size of the video in the referential given by SRDRef
2746 .br
2747 .TP
2748 .B SRDRef (SRDR,v2di, )
2749 .br
2750 Width and Height of the SRD referential
2751 .br
2752 .TP
2753 .B SRDMap (SRDM,uint1, )
2754 .br
2755 Mapping of input videos in reconstructed video, expressed as {Ox,Oy,Ow,Oh,Dx,Dy,Dw,Dh} per input,
2756 .br
2757 * Ox,Oy,Ow,Oh: position and size of the input video (usually matching its SRD property), expressed
2758 .br
2759 * Dx,Dy,Dw,Dh: Position and Size of the input video in the reconstructed output, expressed in the
2760 .br
2761 .TP
2762 .B Alpha (VALP,bool, )
2763 .br
2764 Video in this PID is an alpha map
2765 .br
2766 .TP
2767 .B Mirror (VMIR,uint, )
2768 .br
2769 Mirror mode (as bit mask with flags 0: no mirror, 1: along Y-axis, 2: along X-axis)
2770 .br
2771 .TP
2772 .B Rotate (VROT,uint, )
2773 .br

```

```

2774 Video rotation as value*90 degree anti-clockwise
2775 .br
2776 .TP
2777 .B ClapW (CLPW,frac, )
2778 .br
2779 Width of clean aperture in luma pixels
2780 .br
2781 .TP
2782 .B ClapH (CLPH,frac, )
2783 .br
2784 Height of clean aperture in luma pixels
2785 .br
2786 .TP
2787 .B ClapX (CLPX,frac, )
2788 .br
2789 Horizontal offset of clean aperture center in luma pixels, 0 at image center
2790 .br
2791 .TP
2792 .B ClapY (CLPY,frac, )
2793 .br
2794 Vertical offset of clean aperture center in luma pixels, 0 at image center
2795 .br
2796 .TP
2797 .B NumViews (PNBV,uint, )
2798 .br
2799 Number of views packed in a frame (top-to-bottom only)
2800 .br
2801 .TP
2802 .B Bitrate (RATE,uint, )
2803 .br
2804 Bitrate in bps
2805 .br
2806 .TP
2807 .B Maxrate (MRAT,uint, )
2808 .br
2809 Max bitrate in bps
2810 .br
2811 .TP
2812 .B TargetRate (TBRT,uint, )
2813 .br
2814 Target bitrate in bps, used to setup encoders
2815 .br
2816 .TP
2817 .B DBSize (DBSZ,uint, )
2818 .br
2819 Decode buffer size in bytes
2820 .br
2821 .TP
2822 .B MediaDataSize (MDSZ,luint,D )

```

2823 .br
2824 Size in bytes of media data
2825 .br
2826 .TP
2827 **.B DataRef (DREF,bool,D)**
2828 .br
2829 Data referencing is possible (each compressed frame is a continuous set of bytes in source, with n
2830 .br
2831 .TP
2832 **.B URL (FURL,str,D)**
2833 .br
2834 URL of source
2835 .br
2836 .TP
2837 **.B RemoteURL (RURL,str,D)**
2838 .br
2839 Remote URL of source - used for MPEG-4 systems
2840 .br
2841 .TP
2842 **.B RedirectURL (RELO,str,D)**
2843 .br
2844 Redirection URL of source
2845 .br
2846 .TP
2847 **.B SourcePath (FSRC,str,D)**
2848 .br
2849 Path of source file on file system
2850 .br
2851 .TP
2852 **.B MIMEType (MIME,str,D)**
2853 .br
2854 MIME type of source
2855 .br
2856 .TP
2857 **.B Extension (FEXT,str,D)**
2858 .br
2859 File extension of source
2860 .br
2861 .TP
2862 **.B Cached (CACH,bool,D)**
2863 .br
2864 File is completely cached
2865 .br
2866 .TP
2867 **.B DownloadRate (DLBW,uint,D)**
2868 .br
2869 Download rate of resource in bits per second - changes are signaled through PID info (no reconfigu
2870 .br
2871 .TP

2872 **.B DownloadSize (DLSZ,luint,D)**
2873 **.br**
2874 Size of resource in bytes
2875 **.br**
2876 **.TP**
2877 **.B DownBytes (DLBD,luint,D)**
2878 **.br**
2879 Number of bytes downloaded - changes are signaled through PID info (no reconfigure)
2880 **.br**
2881 **.TP**
2882 **.B ByteRange (FBRA,lfrac,D)**
2883 **.br**
2884 Byte range of resource
2885 **.br**
2886 **.TP**
2887 **.B DisableProgressive (NPRG,uint,)**
2888 **.br**
2889 Some blocks in file need patching (replace or insertion) upon closing, potentially disabling progr
2890 **.br**
2891 **.TP**
2892 **.B IsoAltBrands (ABRD,4cc1,D)**
2893 **.br**
2894 ISOBMFF brands associated with PID/file
2895 **.br**
2896 **.TP**
2897 **.B IsoBrand (MBRD,4cc,D)**
2898 **.br**
2899 ISOBMFF major brand associated with PID/file
2900 **.br**
2901 **.TP**
2902 **.B MovieTime (MHTS,lfrac,D)**
2903 **.br**
2904 ISOBMFF movie header duration and timescale
2905 **.br**
2906 **.TP**
2907 **.B HasSync (PSYN,bool,D)**
2908 **.br**
2909 PID has sync points
2910 **.br**
2911 **.TP**
2912 **.B ServiceWidth (DWDT,uint,D)**
2913 **.br**
2914 Display width of service
2915 **.br**
2916 **.TP**
2917 **.B ServiceHeight (DHGT,uint,D)**
2918 **.br**
2919 Display height of service
2920 **.br**

2921 .TP
2922 **.B CarouselRate** (CARA,uint,D)
2923 .br
2924 Repeat rate in ms for systems carousel data
2925 .br
2926 .TP
2927 **.B AudioVolume** (AVOL,uint,D)
2928 .br
2929 Volume of audio
2930 .br
2931 .TP
2932 **.B AudioPan** (APAN,uint,D)
2933 .br
2934 Balance/Pan of audio
2935 .br
2936 .TP
2937 **.B AudioPriority** (APRI,uint,D)
2938 .br
2939 Audio thread priority
2940 .br
2941 .TP
2942 **.B ProtectionScheme** (SCHT,4cc,)
2943 .br
2944 Protection scheme type (4CC) used
2945 .br
2946 .TP
2947 **.B SchemeVersion** (SCHV,uint,)
2948 .br
2949 Protection scheme version used
2950 .br
2951 .TP
2952 **.B SchemeURI** (SCHU,str,)
2953 .br
2954 Protection scheme URI
2955 .br
2956 .TP
2957 **.B KMS_URI** (KMSU,str,)
2958 .br
2959 URI for key management system
2960 .br
2961 .TP
2962 **.B SelectiveEncryption** (ISSE,bool,)
2963 .br
2964 ISMA/OMA selective encryption is used
2965 .br
2966 .TP
2967 **.B IVLength** (ISIV,uint,)
2968 .br
2969 ISMA IV size

2970 .br
2971 .TP
2972 .B KILength (ISKI,uint,)
2973 .br
2974 ISMA KeyIndication size
2975 .br
2976 .TP
2977 .B CryptType (OMCT,uint,)
2978 .br
2979 OMA encryption type
2980 .br
2981 .TP
2982 .B ContentID (OMID,str,)
2983 .br
2984 OMA Content ID
2985 .br
2986 .TP
2987 .B TextualHeaders (OMTH,str,)
2988 .br
2989 OMA textual headers
2990 .br
2991 .TP
2992 .B PlaintextLen (OMPT,luint,)
2993 .br
2994 OMA size of plaintext data
2995 .br
2996 .TP
2997 .B CryptInfo (ECRI,str,D)
2998 .br
2999 URL (local file only) of crypt info file for this PID, use clear to force passthrough
3000 .br
3001 .TP
3002 .B DecryptInfo (EDRI,str,D)
3003 .br
3004 URL (local file only) of crypt info file for this PID - see decrypter help
3005 .br
3006 .TP
3007 .B SenderNTP (NTPS,luint,DP)
3008 .br
3009 NTP 64 bits timestamp at sender side or grabber side
3010 .br
3011 .TP
3012 .B ReceiverNTP (NTPR,luint,DP)
3013 .br
3014 Receiver NTP (64 bits timestamp) usually associated with the sender NTP property
3015 .br
3016 .TP
3017 .B UTC (UTCD,luint,DP)
3018 .br

```

3019 UTC timestamp (in milliseconds) of parent packet
3020 .br
3021 .TP
3022 .B Encrypted (EPCK,bool, )
3023 .br
3024 Packets for the stream are by default encrypted (however the encryption state is carried in packet
3025 .br
3026 .TP
3027 .B OMAPPreview (ODPR,luint, )
3028 .br
3029 OMA Preview range
3030 .br
3031 .TP
3032 .B CENC_PSSH (PSSH,mem, )
3033 .br
3034 PSSH blob for CENC, formatted as (u32)NbSystems [ (bin128)SystemID(u32)version(u32)KID_count[ (bin
3035 .br
3036 .TP
3037 .B CENC_SAI (SAIS,mem, P)
3038 .br
3039 CENC SAI for the packet, formatted as (char(IV_Size))IV(u16)NbSubSamples [(u16)ClearBytes(u32)Cryp
3040 .br
3041 .TP
3042 .B KeyInfo (CBIV,mem, )
3043 .br
3044 Multi key info formatted as:
3045 .br
3046     is_mkey(u8);
3047 .br
3048     nb_keys(u16);
3049 .br
3050     [
3051 .br
3052         IV_size(u8);
3053 .br
3054         KID(bin128);
3055 .br
3056         if (!IV_size) {;
3057 .br
3058             const_IV_size(u8);
3059 .br
3060             constIV(const_IV_size);
3061 .br
3062         }
3063 .br
3064     ]
3065 .br
3066
3067 .br

```

```

3068 .TP
3069 .B CENCPattern (CPTR,frac, )
3070 .br
3071 CENC crypt pattern, CENC pattern, skip as frac.num crypt as frac.den
3072 .br
3073 .TP
3074 .B CENCStore (CSTR,4cc, )
3075 .br
3076 Storage location 4CC of SAI data
3077 .br
3078 .TP
3079 .B CENCstdMode (CSTM,uint, )
3080 .br
3081 Mode for CENC sample description when using clear samples:
3082 .br
3083 * 0: single sample description is used
3084 .br
3085 * 1: a clear clone of the sample description is created, inserted before the CENC sample descripti
3086 .br
3087 * 2: a clear clone of the sample description is created, inserted after the CENC sample descriptio
3088 .br
3089 .TP
3090 .B AMRModeSet (AMST,uint, )
3091 .br
3092 ModeSet for AMR and AMR-WideBand
3093 .br
3094 .TP
3095 .B SubSampleInfo (SUBS,mem, )
3096 .br
3097 Binary blob describing N subsamples of the sample, formatted as N [(u32)flags(u32)size(u32)codec_p
3098 .br
3099 .TP
3100 .B NALUMaxSize (NALS,uint, )
3101 .br
3102 Max size of NAL units in stream - changes are signaled through PID info change (no reconfigure)
3103 .br
3104 .TP
3105 .B FileNumber (FNUM,uint, P)
3106 .br
3107 Index of file when dumping to files
3108 .br
3109 .TP
3110 .B FileName (FNAM,str, P)
3111 .br
3112 Name of output file when dumping / dashing. Must be set on first packet belonging to new file
3113 .br
3114 .TP
3115 .B IDXName (INAM,str, P)
3116 .br

```


3117 Name of index file when dashing MPEG-2 TS. Must be set on first packet belonging to new file
3118 .br
3119 .TP
3120 **.B FileSuffix (FSUF, str, P)**
3121 .br
3122 File suffix name, replacement for \$FS\$ in tile templates
3123 .br
3124 .TP
3125 **.B EODS (EODS, bool, P)**
3126 .br
3127 End of DASH segment
3128 .br
3129 .TP
3130 **.B CueStart (PCUS, bool, P)**
3131 .br
3132 Set on packets marking the beginning of a DASH/HLS segment for cue-driven segmentation - see dashe
3133 .br
3134 .TP
3135 **.B MediaTime (MTIM, dbl, D)**
3136 .br
3137 Corresponding media time of the parent packet (0 being the origin)
3138 .br
3139 .TP
3140 **.B MaxFrameSize (MFRS, uint, D)**
3141 .br
3142 Max size of frame in stream - changes are signaled through PID info change (no reconfigure)
3143 .br
3144 .TP
3145 **.B AvgFrameSize (AFRS, uint, D)**
3146 .br
3147 Average size of frame in stream (ISOBMFF only, static property)
3148 .br
3149 .TP
3150 **.B MaxTSDelta (MTSD, uint, D)**
3151 .br
3152 Maximum DTS delta between frames (ISOBMFF only, static property)
3153 .br
3154 .TP
3155 **.B MaxCTSOOffset (MCTO, uint, D)**
3156 .br
3157 Maximum absolute CTS offset (ISOBMFF only, static property)
3158 .br
3159 .TP
3160 **.B ConstantDuration (SCTD, uint, D)**
3161 .br
3162 Constant duration of samples, 0 means variable duration (ISOBMFF only, static property)
3163 .br
3164 .TP
3165 **.B TrackTemplate (ITKT, mem, D)**

```
3166 .br
3167 ISOBMFF serialized track box for this PID, without any sample info (empty stbl and empty dref)
3168 .br
3169 .TP
3170 .B TrexTemplate (ITXT,mem,D )
3171 .br
3172 ISOBMFF serialized trex box for this PID
3173 .br
3174 .TP
3175 .B STSDTemplate (ISTD,mem,D )
3176 .br
3177 ISOBMFF serialized sample description box (stsd entry) for this PID
3178 .br
3179 .TP
3180 .B MovieUserData (IMUD,mem,D )
3181 .br
3182 ISOBMFF serialized moov UDTA and other moov-level boxes (list) for this PID
3183 .br
3184 .TP
3185 .B HandlerName (IHDL,str,D )
3186 .br
3187 ISOBMFF track handler name
3188 .br
3189 .TP
3190 .B TrackFlags (ITKF,uint,D )
3191 .br
3192 ISOBMFF track header flags
3193 .br
3194 .TP
3195 .B TrackMatrix (ITKM,sint1,D )
3196 .br
3197 ISOBMFF track header matrix
3198 .br
3199 .TP
3200 .B AltGroup (IALG,uint,D )
3201 .br
3202 ISOBMFF alt group ID
3203 .br
3204 .TP
3205 .B ForceNCTTS (IFNC,bool,D )
3206 .br
3207 ISOBMFF force negative CTS offsets
3208 .br
3209 .TP
3210 .B Disable (ITKD,bool,D )
3211 .br
3212 ISOBMFF disable flag
3213 .br
3214 .TP
```

3215 **.B** Period (PEID,str,D)
3216 .br
3217 ID of DASH period
3218 .br
3219 .TP
3220 **.B** PStart (PEST,lfrac,D)
3221 .br
3222 DASH Period start - cf dasher help
3223 .br
3224 .TP
3225 **.B** PDur (PEDU,lfrac,D)
3226 .br
3227 DASH Period duration - cf dasher help
3228 .br
3229 .TP
3230 **.B** Representation (DRID,str,D)
3231 .br
3232 ID of DASH representation
3233 .br
3234 .TP
3235 **.B** ASID (DAID,uint,D)
3236 .br
3237 ID of parent DASH AS
3238 .br
3239 .TP
3240 **.B** MuxSrc (MSRC,str,D)
3241 .br
3242 Name of mux source(s), set by dasher to direct its outputs
3243 .br
3244 .TP
3245 **.B** DashMode (DMOD,uint,D)
3246 .br
3247 DASH mode to be used by multiplexer if any, set by dasher. 0 is no DASH, 1 is regular DASH, 2 is V
3248 .br
3249 .TP
3250 **.B** DashDur (DDUR,frac,D)
3251 .br
3252 DASH target segment duration in seconds
3253 .br
3254 .TP
3255 **.B** Role (ROLE,str1,D)
3256 .br
3257 List of roles for this PID, where each role string can be a DASH role, a URN:role-value or any oth
3258 .br
3259 .TP
3260 **.B** PDesc (PDES,str1,D)
3261 .br
3262 List of descriptors for the DASH period containing this PID
3263 .br

3264 .TP
3265 .B ASDesc (ACDS,str1,D)
3266 .br
3267 List of conditional descriptors for the DASH AdaptationSet containing this PID. If a PID with the
3268 .br
3269 .TP
3270 .B ASCDesc (AADS,str1,D)
3271 .br
3272 List of common descriptors for the DASH AdaptationSet containing this PID
3273 .br
3274 .TP
3275 .B RDesc (RDES,str1,D)
3276 .br
3277 List of descriptors for the DASH Representation containing this PID
3278 .br
3279 .TP
3280 .B BURL (BURL,str1,D)
3281 .br
3282 List of base URLs for this PID
3283 .br
3284 .TP
3285 .B Template (DTPL,str,)
3286 .br
3287 Template to use for DASH generation for this PID
3288 .br
3289 .TP
3290 .B StartNumber (DRSN,uint,)
3291 .br
3292 Start number to use for this PID - cf dasher help
3293 .br
3294 .TP
3295 .B xlink (XLNK,str,D)
3296 .br
3297 Remote period URL for DASH - cf dasher help
3298 .br
3299 .TP
3300 .B ClampDur (DCMD,lfrac,D)
3301 .br
3302 Max media duration to process from PID in DASH mode
3303 .br
3304 .TP
3305 .B HLSPL (HLVP,str,D)
3306 .br
3307 Name of the HLS variant playlist for this media
3308 .br
3309 .TP
3310 .B HLSGroup (HLGI,str,D)
3311 .br
3312 Name of HLS Group of a stream

3313 .br
3314 .TP
3315 **.B** HLSMExt (HLMX,str1,D)
3316 .br
3317 List of extensions to add to the master playlist for this PID
3318 .br
3319 .TP
3320 **.B** HLSVExt (HLVX,str1,D)
3321 .br
3322 List of extensions to add to the variant playlist for this PID
3323 .br
3324 .TP
3325 **.B** DCue (DCUE,str,D)
3326 .br
3327 Name of a cue list file for this PID - see dasher help
3328 .br
3329 .TP
3330 **.B** DSegs (DCNS,uint,D)
3331 .br
3332 Number of DASH segments defined by the DASH cue info
3333 .br
3334 .TP
3335 **.B** Codec (CODS,str,D)
3336 .br
3337 codec parameter string to force. If starting with '.', appended to ISOBMFF code point; otherwise r
3338 .br
3339 .TP
3340 **.B** SingleScale (DSTS,bool,D)
3341 .br
3342 Movie header should use the media timescale of the first track added
3343 .br
3344 .TP
3345 **.B** RequireReorder (PUDP,bool,D)
3346 .br
3347 PID packets come from source with losses and reordering happening (UDP)
3348 .br
3349 .TP
3350 **.B** Primary (PITM,bool,D)
3351 .br
3352 Primary item in ISOBMFF
3353 .br
3354 .TP
3355 **.B** DFMode (DFWD,uint,D)
3356 .br
3357 DASH forward mode is used for this PID. If 2, the manifest is also carried in packet property
3358 .br
3359 .TP
3360 **.B** DFManifest (DMPD,str,D)
3361 .br

3362 Value of manifest in forward mode
3363 .br
3364 .TP
3365 .B DFVariant (DHLV,str1,D)
3366 .br
3367 Value of variant playlist in forward mode
3368 .br
3369 .TP
3370 .B DFVariantName (DHLN,str1,D)
3371 .br
3372 Value of variant playlist name in forward mode
3373 .br
3374 .TP
3375 .B DFPStart (DPST,luint,D)
3376 .br
3377 Value of active period start time in forward mode
3378 .br
3379 .TP
3380 .B HLSKey (HLSK,str,)
3381 .br
3382 URI, KEYFORMAT and KEYFORMATVERSIONS for HLS full segment encryption creation, Key URI otherwise (
3383 .br
3384 .TP
3385 .B HLSIV (HLSI,mem,)
3386 .br
3387 Init Vector for HLS decode
3388 .br
3389 .TP
3390 .B ColorPrimaries (CPRM,cprm,D)
3391 .br
3392 Color primaries
3393 .br
3394 .TP
3395 .B ColorTransfer (CTRC,ctfc,D)
3396 .br
3397 Color transfer characteristics
3398 .br
3399 .TP
3400 .B ColorMatrix (CMXC,cmxc,D)
3401 .br
3402 Color matrix coefficient
3403 .br
3404 .TP
3405 .B FullRange (CFRA,bool,D)
3406 .br
3407 Color full range flag
3408 .br
3409 .TP
3410 .B Chroma (CFMT,uint,D)

3411 .br
3412 Chroma format (see ISO/IEC 23001-8 / 23091-2)
3413 .br
3414 .TP
3415 **.B ChromaLoc** (CLOC,uint,D)
3416 .br
3417 Chroma location (see ISO/IEC 23001-8 / 23091-2)
3418 .br
3419 .TP
3420 **.B ContentLightLevel** (CLLI,mem,D)
3421 .br
3422 Content light level, payload of clii box (see ISO/IEC 14496-12), can be set as a list of 2 integer
3423 .br
3424 .TP
3425 **.B MasterDisplayColour** (MDCV,mem,D)
3426 .br
3427 Master display colour info, payload of mdcv box (see ISO/IEC 14496-12), can be set as a list of 10
3428 .br
3429 .TP
3430 **.B SrcMagic** (PSMG,luint,D)
3431 .br
3432 Magic number to store in the track, only used by importers
3433 .br
3434 .TP
3435 **.B MuxIndex** (TIDX,luint,D)
3436 .br
3437 Target track index in destination file, stored by lowest value first (not set by demultiplexers)
3438 .br
3439 .TP
3440 **.B NoTSLoop** (NTSL,bool,)
3441 .br
3442 Timestamps on this PID are adjusted in case of loops (used by TS multiplexer output)
3443 .br
3444 .TP
3445 **.B MHAPProfiles** (MHCP,uint1,D)
3446 .br
3447 List of compatible profiles for this MPEG-H Audio object
3448 .br
3449 .TP
3450 **.B FragStart** (PFRB,uint,DP)
3451 .br
3452 Packet is a fragment start (value 1) or a segment start (value 2)
3453 .br
3454 .TP
3455 **.B FragRange** (PFRR,lfrac,DP)
3456 .br
3457 Start and end position in bytes of fragment if packet is a fragment or segment start
3458 .br
3459 .TP

```
3460 .B SIDXRange (PFSR,lfrac,DP)
3461 .br
3462 Start and end position in bytes of sidx if packet is a fragment or segment start
3463 .br
3464 .TP
3465 .B MoofTemplate (MFTP,mem,DP)
3466 .br
3467 Serialized moof box corresponding to the start of a movie fragment or segment (with styp and optio
3468 .br
3469 .TP
3470 .B InitSeg (PCKI,bool, P)
3471 .br
3472 Set to true if packet is a complete DASH init segment file
3473 .br
3474 .TP
3475 .B RawGrab (PGRB,uint,D )
3476 .br
3477 PID is a raw media grabber (webcam, microphone, etc...). Value 2 is used for front camera
3478 .br
3479 .TP
3480 .B KeepAfterEOS (PKAE,bool,D )
3481 .br
3482 PID must be kept alive after EOS (LASEr and BIFS)
3483 .br
3484 .TP
3485 .B CoverArt (PCOV,mem,D )
3486 .br
3487 PID cover art image data. If associated data is NULL, the data is carried in the PID
3488 .br
3489 .TP
3490 .B BufferLength (PBPL,uint,D )
3491 .br
3492 Playout buffer in ms
3493 .br
3494 .TP
3495 .B MaxBuffer (PBMX,uint,D )
3496 .br
3497 Maximum buffer occupancy in ms
3498 .br
3499 .TP
3500 .B ReBuffer (PBRE,uint,D )
3501 .br
3502 Rebuffer threshold in ms, 0 disable rebuffering
3503 .br
3504 .TP
3505 .B ViewIdx (VIDX,uint,D )
3506 .br
3507 View index for multiview (1 being left)
3508 .br
```


3509 .TP
3510 .B FragURL (OFRA,str,D)
3511 .br
3512 Fragment URL (without '#') of original URL (used by some filters to set the property on media PIDs
3513 .br
3514 .TP
3515 .B ROUTEIP (RSIP,str,D)
3516 .br
3517 ROUTE session IP address
3518 .br
3519 .TP
3520 .B ROUTEPort (RSPN,uint,D)
3521 .br
3522 ROUTE session port number
3523 .br
3524 .TP
3525 .B ROUTEName (RSFN,str,D)
3526 .br
3527 Name (location) of raw file to advertise in ROUTE session
3528 .br
3529 .TP
3530 .B ROUTECarousel (RSCR,frac,D)
3531 .br
3532 Carousel period in seconds of raw file in ROUTE session
3533 .br
3534 .TP
3535 .B ROUTEUpload (RSST,frac,D)
3536 .br
3537 Upload time in seconds of raw file in ROUTE session
3538 .br
3539 .TP
3540 .B Stereo (PSTT,uint,D)
3541 .br
3542 Stereo type of video
3543 .br
3544 .TP
3545 .B Projection (PPJT,uint,D)
3546 .br
3547 Projection type of video
3548 .br
3549 .TP
3550 .B InitalPose (PPOS,v3di,D)
3551 .br
3552 Initial pose for 360 video, in degrees expressed as 16.16 bits (x is yaw, y is pitch, z is roll)
3553 .br
3554 .TP
3555 .B CMPad (PCMP,uint,D)
3556 .br
3557 Number of pixels to pad from edge of each face in cube map

```
3558 .br
3559 .TP
3560 .B EQRClamp (PEQC,v4di,D )
3561 .br
3562 Clamping of frame for EQR as 0.32 fixed point (x is top, y is bottom, z is left and w is right)
3563 .br
3564 .TP
3565 .B SceneNode (PSND,bool, )
3566 .br
3567 PID is a scene node decoder (AFX BitWrapper in BIFS)
3568 .br
3569 .TP
3570 .B OrigCryptoScheme (POCS,uint, )
3571 .br
3572 Original crypto scheme on a decrypted PID
3573 .br
3574 .TP
3575 .B TSBSegs (PTSN,uint,D )
3576 .br
3577 Time shift in number of segments for HAS streams, only set by dashin and dasher filters
3578 .br
3579 .TP
3580 .B IsManifest (PHSM,bool,D )
3581 .br
3582 PID is a HAS manifest
3583 .br
3584 .TP
3585 .B Sparse (PSPA,bool,D )
3586 .br
3587 PID has potentially empty times between packets
3588 .br
3589 .TP
3590 .B SkipBegin (PCKS,uint, P)
3591 .br
3592 Amount of media to skip from beginning of packet in PID timescale
3593 .br
3594 .TP
3595 .B SkipPres (PCKD,bool, P)
3596 .br
3597 Packet and any following with CTS greater than this packet shall not be presented (used by reframe
3598 .br
3599 .TP
3600 .B HLSRef (HPLR,luint,DP)
3601 .br
3602 HLS playlist reference, gives a unique ID identifying media mux, and indicated in packets carrying
3603 .br
3604 .TP
3605 .B LLHLS (HLST,uint,D )
3606 .br
```

```
3607 HLS low latency mode
3608 .br
3609 .TP
3610 .B LLHLSFragNum (HLSN,uint, P)
3611 .br
3612 LLHLS fragment number
3613 .br
3614 .TP
3615 .B DownloadSession (GHTT,ptr,D )
3616 .br
3617 Pointer to download session
3618 .br
3619 .TP
3620 .B HasTemi (PTem,bool,D )
3621 .br
3622 TEMI present flag
3623 .br
3624 .TP
3625 .B XPSMask (PXPM,uint,DP)
3626 .br
3627 Parameter set mask
3628 .br
3629 .TP
3630 .B RangeEnd (PCER,bool, P)
3631 .br
3632 Signal packet is the last in the desired play range
3633 .br
3634 .SH Pixel formats
3635 .LP
3636 .br
3637 .TP
3638 .B yuv420 (ext *.yuv)
3639 .br
3640 Planar YUV 420 8 bit
3641 .br
3642 .TP
3643 .B yvu420 (ext *.yvu)
3644 .br
3645 Planar YVU 420 8 bit
3646 .br
3647 .TP
3648 .B yuv420_10 (ext *.yuv1)
3649 .br
3650 Planar YUV 420 10 bit
3651 .br
3652 .TP
3653 .B yuv422 (ext *.yuv2)
3654 .br
3655 Planar YUV 422 8 bit
```

```
3656 .br
3657 .TP
3658 .B yuv422_10 (ext *.yp21)
3659 .br
3660 Planar YUV 422 10 bit
3661 .br
3662 .TP
3663 .B yuv444 (ext *.yuv4)
3664 .br
3665 Planar YUV 444 8 bit
3666 .br
3667 .TP
3668 .B yuv444_10 (ext *.yp41)
3669 .br
3670 Planar YUV 444 10 bit
3671 .br
3672 .TP
3673 .B uyvy (ext *.uyvy)
3674 .br
3675 Packed UYVY 422 8 bit
3676 .br
3677 .TP
3678 .B vyuy (ext *.vyuy)
3679 .br
3680 Packed VYUV 422 8 bit
3681 .br
3682 .TP
3683 .B yuyv (ext *.yuyv)
3684 .br
3685 Packed YUYV 422 8 bit
3686 .br
3687 .TP
3688 .B yvyu (ext *.yvyu)
3689 .br
3690 Packed YVYU 422 8 bit
3691 .br
3692 .TP
3693 .B uyvl (ext *.uyvl)
3694 .br
3695 Packed UYVY 422 10->16 bit
3696 .br
3697 .TP
3698 .B vyul (ext *.vyul)
3699 .br
3700 Packed VYUV 422 10->16 bit
3701 .br
3702 .TP
3703 .B yuy1 (ext *.yuy1)
3704 .br
```

```
3705 Packed YUYV 422 10->16 bit
3706 .br
3707 .TP
3708 .B yvy1 (ext *.yvy1)
3709 .br
3710 Packed YVYU 422 10->16 bit
3711 .br
3712 .TP
3713 .B nv12 (ext *.nv12)
3714 .br
3715 Semi-planar YUV 420 8 bit, Y plane and UV packed plane
3716 .br
3717 .TP
3718 .B nv21 (ext *.nv21)
3719 .br
3720 Semi-planar YVU 420 8 bit, Y plane and VU packed plane
3721 .br
3722 .TP
3723 .B nv11 (ext *.nv11)
3724 .br
3725 Semi-planar YUV 420 10 bit, Y plane and UV plane
3726 .br
3727 .TP
3728 .B nv21 (ext *.nv21)
3729 .br
3730 Semi-planar YVU 420 8 bit, Y plane and VU plane
3731 .br
3732 .TP
3733 .B yuva (ext *.yuva)
3734 .br
3735 Planar YUV+alpha 420 8 bit
3736 .br
3737 .TP
3738 .B yuvd (ext *.yuvd)
3739 .br
3740 Planar YUV+depth 420 8 bit
3741 .br
3742 .TP
3743 .B yuv444a (ext *.yp4a)
3744 .br
3745 Planar YUV+alpha 444 8 bit
3746 .br
3747 .TP
3748 .B yuv444p (ext *.yv4p)
3749 .br
3750 Packed YUV 444 8 bit
3751 .br
3752 .TP
3753 .B v308 (ext *.v308)
```

```
3754 .br
3755 Packed VYU 444 8 bit
3756 .br
3757 .TP
3758 .B yuv444ap (ext *.y4ap)
3759 .br
3760 Packed YUV+alpha 444 8 bit
3761 .br
3762 .TP
3763 .B v408 (ext *.v408)
3764 .br
3765 Packed UYV+alpha 444 8 bit
3766 .br
3767 .TP
3768 .B v410 (ext *.v410)
3769 .br
3770 Packed UYV 444 10 bit LE
3771 .br
3772 .TP
3773 .B v210 (ext *.v210)
3774 .br
3775 Packed UYVY 422 10 bit LE
3776 .br
3777 .TP
3778 .B grey (ext *.grey)
3779 .br
3780 Greyscale 8 bit
3781 .br
3782 .TP
3783 .B algr (ext *.algr)
3784 .br
3785 Alpha+Grey 8 bit
3786 .br
3787 .TP
3788 .B gral (ext *.gral)
3789 .br
3790 Grey+Alpha 8 bit
3791 .br
3792 .TP
3793 .B rgb4 (ext *.rgb4)
3794 .br
3795 RGB 444, 12 bits (16 stored) / pixel
3796 .br
3797 .TP
3798 .B rgb5 (ext *.rgb5)
3799 .br
3800 RGB 555, 15 bits (16 stored) / pixel
3801 .br
3802 .TP
```

```
3803 .B rgb6 (ext *.rgb6)
3804 .br
3805 RGB 555, 16 bits / pixel
3806 .br
3807 .TP
3808 .B rgba (ext *.rgba)
3809 .br
3810 RGBA 32 bits (8 bits / component)
3811 .br
3812 .TP
3813 .B argb (ext *.argb)
3814 .br
3815 ARGB 32 bits (8 bits / component)
3816 .br
3817 .TP
3818 .B bgra (ext *.bgra)
3819 .br
3820 BGRA 32 bits (8 bits / component)
3821 .br
3822 .TP
3823 .B abgr (ext *.abgr)
3824 .br
3825 ABGR 32 bits (8 bits / component)
3826 .br
3827 .TP
3828 .B rgb (ext *.rgb)
3829 .br
3830 RGB 24 bits (8 bits / component)
3831 .br
3832 .TP
3833 .B bgr (ext *.bgr)
3834 .br
3835 BGR 24 bits (8 bits / component)
3836 .br
3837 .TP
3838 .B xrgb (ext *.xrgb)
3839 .br
3840 xRGB 32 bits (8 bits / component)
3841 .br
3842 .TP
3843 .B rgbx (ext *.rgbx)
3844 .br
3845 RGBx 32 bits (8 bits / component)
3846 .br
3847 .TP
3848 .B xbgr (ext *.xbgr)
3849 .br
3850 xBGR 32 bits (8 bits / component)
3851 .br
```

3852 .TP
3853 **.B bgrx** (ext *.bgrx)
3854 .br
3855 BGRx 32 bits (8 bits / component)
3856 .br
3857 .TP
3858 **.B rgbd** (ext *.rgbd)
3859 .br
3860 RGB+depth 32 bits (8 bits / component)
3861 .br
3862 .TP
3863 **.B rgbds** (ext *.rgbds)
3864 .br
3865 RGB+depth+bit shape (8 bits / RGB component, 7 bit depth (low bits) + 1 bit shape)
3866 .br
3867 .TP
3868 **.B rgbs** (ext *.rgbs)
3869 .br
3870 RGB 24 bits stereo (side-by-side) - to be removed
3871 .br
3872
3873 .br
3874 .TP
3875 **.B rgbas** (ext *.rgbas)
3876 .br
3877 RGBA 32 bits stereo (side-by-side) - to be removed
3878 .br
3879
3880 .br
3881 .TP
3882 **.B extgl** (ext *.extgl)
3883 .br
3884 External OpenGL texture of unknown format, to be used with samplerExternalOES
3885 .br
3886
3887 .br
3888 **.SH Audio formats**
3889 .LP
3890 .br
3891 .TP
3892 **.B u8** (ext *.pc8)
3893 .br
3894 8 bit PCM
3895 .br
3896 .TP
3897 **.B s16** (ext *.pcm)
3898 .br
3899 16 bit PCM Little Endian
3900 .br


```
3901 .TP
3902 .B s16b (ext *.pcmb)
3903 .br
3904 16 bit PCM Big Endian
3905 .br
3906 .TP
3907 .B s24 (ext *.s24)
3908 .br
3909 24 bit PCM
3910 .br
3911 .TP
3912 .B s32 (ext *.s32)
3913 .br
3914 32 bit PCM Little Endian
3915 .br
3916 .TP
3917 .B flt (ext *.flt)
3918 .br
3919 32-bit floating point PCM
3920 .br
3921 .TP
3922 .B dbl (ext *.dbl)
3923 .br
3924 64-bit floating point PCM
3925 .br
3926 .TP
3927 .B u8p (ext *.pc8p)
3928 .br
3929 8 bit PCM planar
3930 .br
3931 .TP
3932 .B s16p (ext *.pcmp)
3933 .br
3934 16 bit PCM Little Endian planar
3935 .br
3936 .TP
3937 .B s24p (ext *.s24p)
3938 .br
3939 24 bit PCM planar
3940 .br
3941 .TP
3942 .B s32p (ext *.s32p)
3943 .br
3944 32 bit PCM Little Endian planar
3945 .br
3946 .TP
3947 .B fltp (ext *.fltp)
3948 .br
3949 32-bit floating point PCM planar
```

3950 .br
3951 .TP
3952 **.B dblp (ext *.dblp)**
3953 .br
3954 64-bit floating point PCM planar
3955 .br
3956 **.SH Stream types**
3957 .LP
3958 .br
3959 .TP
3960 **.B Visual**
3961 .br
3962 Video or Image stream
3963 .br
3964 .TP
3965 **.B Audio**
3966 .br
3967 Audio stream
3968 .br
3969 .TP
3970 **.B SceneDescription**
3971 .br
3972 Scene stream
3973 .br
3974 .TP
3975 **.B Text**
3976 .br
3977 Text or subtitle stream
3978 .br
3979 .TP
3980 **.B Metadata**
3981 .br
3982 Metadata stream
3983 .br
3984 .TP
3985 **.B File**
3986 .br
3987 Raw file stream
3988 .br
3989 .TP
3990 **.B Encrypted**
3991 .br
3992 Encrypted media stream
3993 .br
3994 .TP
3995 **.B ObjectDescriptor**
3996 .br
3997 MPEG-4 ObjectDescriptor stream
3998 .br

3999 .TP
4000 **.B ClockReference**
4001 .br
4002 MPEG-4 Clock Reference stream
4003 .br
4004 .TP
4005 **.B MPEG7**
4006 .br
4007 MPEG-7 description stream
4008 .br
4009 .TP
4010 **.B IPMP**
4011 .br
4012 MPEG-4 IPMP/DRM stream
4013 .br
4014 .TP
4015 **.B OCI**
4016 .br
4017 MPEG-4 ObjectContentInformation stream
4018 .br
4019 .TP
4020 **.B MPEGJ**
4021 .br
4022 MPEG-4 JAVA stream
4023 .br
4024 .TP
4025 **.B Interaction**
4026 .br
4027 MPEG-4 Interaction Sensor stream
4028 .br
4029 .TP
4030 **.B Font**
4031 .br
4032 MPEG-4 Font stream
4033 .br
4034 **.SH Codecs**
4035 .LP
4036 .br
4037 .TP
4038 **.B bifs**
4039 .br
4040 MPEG-4 BIFS v1 Scene Description
4041 .br
4042 .TP
4043 **.B bifs2**
4044 .br
4045 MPEG-4 BIFS v2 Scene Description
4046 .br
4047 .TP

4048 **.B bifsX**
4049 **.br**
4050 MPEG-4 BIFS Extended Scene Description
4051 **.br**
4052 **.TP**
4053 **.B od**
4054 **.br**
4055 MPEG-4 ObjectDescriptor v1
4056 **.br**
4057 **.TP**
4058 **.B od2**
4059 **.br**
4060 MPEG-4 ObjectDescriptor v2
4061 **.br**
4062 **.TP**
4063 **.B interact**
4064 **.br**
4065 MPEG-4 Interaction Stream
4066 **.br**
4067 **.TP**
4068 **.B afx**
4069 **.br**
4070 MPEG-4 AFX Stream
4071 **.br**
4072 **.TP**
4073 **.B font**
4074 **.br**
4075 MPEG-4 Font Stream
4076 **.br**
4077 **.TP**
4078 **.B syntax**
4079 **.br**
4080 MPEG-4 Synthetized Texture
4081 **.br**
4082 **.TP**
4083 **.B m4txt**
4084 **.br**
4085 MPEG-4 Streaming Text
4086 **.br**
4087 **.TP**
4088 **.B laser**
4089 **.br**
4090 MPEG-4 LAsER
4091 **.br**
4092 **.TP**
4093 **.B saf**
4094 **.br**
4095 MPEG-4 Simple Aggregation Format
4096 **.br**

4097 .TP
4098 .B cmp|m4ve|m4v
4099 .br
4100 MPEG-4 Visual part 2
4101 .br
4102 .TP
4103 .B 264|avc|h264
4104 .br
4105 MPEG-4 AVC|H264 Video
4106 .br
4107 .TP
4108 .B avcps
4109 .br
4110 MPEG-4 AVC|H264 Video Parameter Sets
4111 .br
4112 .TP
4113 .B svc|avc|264|h264
4114 .br
4115 MPEG-4 AVC|H264 Scalable Video Coding
4116 .br
4117 .TP
4118 .B mvc
4119 .br
4120 MPEG-4 AVC|H264 Multiview Video Coding
4121 .br
4122 .TP
4123 .B hvc|hevc|h265
4124 .br
4125 HEVC Video
4126 .br
4127 .TP
4128 .B lhvc|shvc|mhvc
4129 .br
4130 HEVC Video Layered Extensions
4131 .br
4132 .TP
4133 .B m2vs
4134 .br
4135 MPEG-2 Visual Simple
4136 .br
4137 .TP
4138 .B m2v
4139 .br
4140 MPEG-2 Visual Main
4141 .br
4142 .TP
4143 .B m2v|m2vsnr
4144 .br
4145 MPEG-2 Visual SNR

4146 .br
4147 .TP
4148 .B m2v|m2vspat
4149 .br
4150 MPEG-2 Visual Spatial
4151 .br
4152 .TP
4153 .B m2v|m2vh
4154 .br
4155 MPEG-2 Visual High
4156 .br
4157 .TP
4158 .B m2v|m2v4
4159 .br
4160 MPEG-2 Visual 422
4161 .br
4162 .TP
4163 .B m1v
4164 .br
4165 MPEG-1 Video
4166 .br
4167 .TP
4168 .B jpg|jpeg
4169 .br
4170 JPEG Image
4171 .br
4172 .TP
4173 .B png
4174 .br
4175 PNG Image
4176 .br
4177 .TP
4178 .B jp2|j2k
4179 .br
4180 JPEG2000 Image
4181 .br
4182 .TP
4183 .B aac
4184 .br
4185 MPEG-4 AAC Audio
4186 .br
4187 .TP
4188 .B aac|aac2m
4189 .br
4190 MPEG-2 AAC Audio Main
4191 .br
4192 .TP
4193 .B aac|aac2l
4194 .br

4195 MPEG-2 AAC Audio Low Complexity
4196 .br
4197 .TP
4198 .B aac|aac2s
4199 .br
4200 MPEG-2 AAC Audio Scalable Sampling Rate
4201 .br
4202 .TP
4203 .B mp3|m1a
4204 .br
4205 MPEG-1 Audio
4206 .br
4207 .TP
4208 .B mp2
4209 .br
4210 MPEG-2 Audio
4211 .br
4212 .TP
4213 .B mp1
4214 .br
4215 MPEG-1 Audio Layer 1
4216 .br
4217 .TP
4218 .B h263
4219 .br
4220 H263 Video
4221 .br
4222 .TP
4223 .B h263
4224 .br
4225 H263 Video
4226 .br
4227 .TP
4228 .B hvt1
4229 .br
4230 HEVC tiles Video
4231 .br
4232 .TP
4233 .B evc|evrc
4234 .br
4235 EVRC Voice
4236 .br
4237 .TP
4238 .B smv
4239 .br
4240 SMV Voice
4241 .br
4242 .TP
4243 .B qcp|qcelp

4244	.br
4245	QCELP Voice
4246	.br
4247	.TP
4248	.B amr
4249	.br
4250	AMR Audio
4251	.br
4252	.TP
4253	.B amr amrwb
4254	.br
4255	AMR WideBand Audio
4256	.br
4257	.TP
4258	.B qcp evrcpv
4259	.br
4260	EVRC (PacketVideo MUX) Audio
4261	.br
4262	.TP
4263	.B vc1
4264	.br
4265	SMPTE VC-1 Video
4266	.br
4267	.TP
4268	.B dirac
4269	.br
4270	Dirac Video
4271	.br
4272	.TP
4273	.B ac3
4274	.br
4275	AC3 Audio
4276	.br
4277	.TP
4278	.B eac3
4279	.br
4280	Enhanced AC3 Audio
4281	.br
4282	.TP
4283	.B mlp
4284	.br
4285	Dolby TrueHD
4286	.br
4287	.TP
4288	.B dra
4289	.br
4290	DRA Audio
4291	.br
4292	.TP

4293	.B g719
4294	.br
4295	G719 Audio
4296	.br
4297	.TP
4298	.B dstca
4299	.br
4300	DTS Coherent Acoustics Audio
4301	.br
4302	.TP
4303	.B dtsh
4304	.br
4305	DTS-HD High Resolution Audio
4306	.br
4307	.TP
4308	.B dstm
4309	.br
4310	DTS-HD Master Audio
4311	.br
4312	.TP
4313	.B dts1
4314	.br
4315	DTS Express low bit rate Audio
4316	.br
4317	.TP
4318	.B opus
4319	.br
4320	Opus Audio
4321	.br
4322	.TP
4323	.B eti
4324	.br
4325	DVB Event Information
4326	.br
4327	.TP
4328	.B svgr
4329	.br
4330	SVG over RTP
4331	.br
4332	.TP
4333	.B svgzr
4334	.br
4335	SVG+gz over RTP
4336	.br
4337	.TP
4338	.B dims
4339	.br
4340	3GPP DIMS Scene
4341	.br

4342 .TP
4343 .B vtt
4344 .br
4345 WebVTT Text
4346 .br
4347 .TP
4348 .B txt
4349 .br
4350 Simple Text Stream
4351 .br
4352 .TP
4353 .B txtxt
4354 .br
4355 Metadata Text Stream
4356 .br
4357 .TP
4358 .B mxml
4359 .br
4360 Metadata XML Stream
4361 .br
4362 .TP
4363 .B subs
4364 .br
4365 Subtitle text Stream
4366 .br
4367 .TP
4368 .B subx
4369 .br
4370 Subtitle XML Stream
4371 .br
4372 .TP
4373 .B tx3g
4374 .br
4375 Subtitle/text 3GPP/Apple Stream
4376 .br
4377 .TP
4378 .B ssa
4379 .br
4380 SSA /ASS Subtitles
4381 .br
4382 .TP
4383 .B theo|theora
4384 .br
4385 Theora Video
4386 .br
4387 .TP
4388 .B vorb|vorbis
4389 .br
4390 Vorbis Audio

4391 .br
4392 .TP
4393 **.B opus**
4394 .br
4395 Opus Audio
4396 .br
4397 .TP
4398 **.B flac**
4399 .br
4400 Flac Audio
4401 .br
4402 .TP
4403 **.B spx|speex**
4404 .br
4405 Speex Audio
4406 .br
4407 .TP
4408 **.B vobsub**
4409 .br
4410 VobSub Subtitle
4411 .br
4412 .TP
4413 **.B vobsub**
4414 .br
4415 VobSub Subtitle
4416 .br
4417 .TP
4418 **.B adpcm**
4419 .br
4420 AD-PCM
4421 .br
4422 .TP
4423 **.B csvd**
4424 .br
4425 IBM CSVD
4426 .br
4427 .TP
4428 **.B alaw**
4429 .br
4430 ALAW
4431 .br
4432 .TP
4433 **.B mulaw**
4434 .br
4435 MULAW
4436 .br
4437 .TP
4438 **.B okiadpcm**
4439 .br

4440	OKI ADPCM
4441	.br
4442	.TP
4443	.B dviadpcm
4444	.br
4445	DVI ADPCM
4446	.br
4447	.TP
4448	.B digistd
4449	.br
4450	DIGISTD
4451	.br
4452	.TP
4453	.B yamadpcm
4454	.br
4455	YAMAHA ADPCM
4456	.br
4457	.TP
4458	.B truespeech
4459	.br
4460	DSP TrueSpeech
4461	.br
4462	.TP
4463	.B g610
4464	.br
4465	GSM 610
4466	.br
4467	.TP
4468	.B imulaw
4469	.br
4470	IBM MULAW
4471	.br
4472	.TP
4473	.B ialaw
4474	.br
4475	IBM ALAW
4476	.br
4477	.TP
4478	.B iadpcl
4479	.br
4480	IBM ADPCL
4481	.br
4482	.TP
4483	.B swf
4484	.br
4485	Adobe Flash
4486	.br
4487	.TP
4488	.B raw

4489	.br
4490	Raw media
4491	.br
4492	.TP
4493	.B av1 ivf obu av1b
4494	.br
4495	AOM AV1 Video
4496	.br
4497	.TP
4498	.B vp8 ivf
4499	.br
4500	VP8 Video
4501	.br
4502	.TP
4503	.B vp9 ivf
4504	.br
4505	VP9 Video
4506	.br
4507	.TP
4508	.B vp10 ivf
4509	.br
4510	VP10 Video
4511	.br
4512	.TP
4513	.B mhas
4514	.br
4515	MPEG-H Audio
4516	.br
4517	.TP
4518	.B mhas
4519	.br
4520	MPEG-H AudioMux
4521	.br
4522	.TP
4523	.B prores apch
4524	.br
4525	ProRes Video 422 HQ
4526	.br
4527	.TP
4528	.B prores apco
4529	.br
4530	ProRes Video 422 Proxy
4531	.br
4532	.TP
4533	.B prores apcn
4534	.br
4535	ProRes Video 422 STD
4536	.br
4537	.TP

4538 **.B prores|apcs**
4539 **.br**
4540 ProRes Video 422 LT
4541 **.br**
4542 **.TP**
4543 **.B prores|ap4x**
4544 **.br**
4545 ProRes Video 4444 XQ
4546 **.br**
4547 **.TP**
4548 **.B prores|ap4h**
4549 **.br**
4550 ProRes Video 4444
4551 **.br**
4552 **.TP**
4553 **.B ffmpeg**
4554 **.br**
4555 FFmpeg unmapped codec
4556 **.br**
4557 **.TP**
4558 **.B tmcd**
4559 **.br**
4560 QT TimeCode
4561 **.br**
4562 **.TP**
4563 **.B vvc|266|h266**
4564 **.br**
4565 VVC Video
4566 **.br**
4567 **.TP**
4568 **.B vvs1**
4569 **.br**
4570 VVC Subpicture Video
4571 **.br**
4572 **.TP**
4573 **.B usac|xheaac**
4574 **.br**
4575 xHEAAC / USAC Audio
4576 **.br**
4577 **.TP**
4578 **.B ffv1**
4579 **.br**
4580 FFmpeg Video Codec 1
4581 **.br**
4582 **.TP**
4583 **.B dvbs**
4584 **.br**
4585 DVB Subtitles
4586 **.br**

4587 .TP
4588 .B dvbs
4589 .br
4590 DVB-TeleText
4591 .br
4592 .SH Stream types
4593 .LP
4594 .br
4595 .TP
4596 .B mono (int 1)
4597 .br
4598 Layout 0x0000000000000004
4599 .br
4600 .TP
4601 .B stereo (int 2)
4602 .br
4603 Layout 0x0000000000000003
4604 .br
4605 .TP
4606 .B 3/0.0 (int 3)
4607 .br
4608 Layout 0x0000000000000007
4609 .br
4610 .TP
4611 .B 3/1.0 (int 4)
4612 .br
4613 Layout 0x0000000000000407
4614 .br
4615 .TP
4616 .B 3/2.0 (int 5)
4617 .br
4618 Layout 0x0000000000000307
4619 .br
4620 .TP
4621 .B 3/2.1 (int 6)
4622 .br
4623 Layout 0x000000000000030f
4624 .br
4625 .TP
4626 .B 5/2.1 (int 7)
4627 .br
4628 Layout 0x000000000000030f
4629 .br
4630 .TP
4631 .B 1+1 (int 8)
4632 .br
4633 Layout 0x0000000000000003
4634 .br
4635 .TP

4636 .B 2/1.0 (int 9)
4637 .br
4638 Layout 0x0000000000000403
4639 .br
4640 .TP
4641 .B 2/2.0 (int 10)
4642 .br
4643 Layout 0x000000000000033
4644 .br
4645 .TP
4646 .B 3/3.1 (int 11)
4647 .br
4648 Layout 0x00000000000043f
4649 .br
4650 .TP
4651 .B 3/4.1 (int 12)
4652 .br
4653 Layout 0x00000000000033f
4654 .br
4655 .TP
4656 .B 11/11.2 (int 13)
4657 .br
4658 Layout 0x000000003ffe67cf
4659 .br
4660 .TP
4661 .B 5/2.1 (int 14)
4662 .br
4663 Layout 0x00000000006030f
4664 .br
4665 .TP
4666 .B 5/5.2 (int 15)
4667 .br
4668 Layout 0x00000000606630f
4669 .br
4670 .TP
4671 .B 5/4.1 (int 16)
4672 .br
4673 Layout 0x0000000036003f
4674 .br
4675 .TP
4676 .B 6/5.1 (int 17)
4677 .br
4678 Layout 0x0000000023e003f
4679 .br
4680 .TP
4681 .B 6/7.1 (int 18)
4682 .br
4683 Layout 0x0000600023e003f
4684 .br

4685 .TP
4686 .B 5/6.1 (int 19)
4687 .br
4688 Layout 0x00000000036630f
4689 .br
4690 .TP
4691 .B 7/6.1 (int 20)
4692 .br
4693 Layout 0x000000600036630f
4694 .br
4695 .SH EXAMPLES
4696 .TP
4697 Basic and advanced examples are available at <https://wiki.gpac.io/Filters>
4698 .SH MORE
4699 .LP
4700 Authors: GPAC developers, see git repo history (-log)
4701 .br
4702 For bug reports, feature requests, more information and source code, visit <https://github.com/gpac>
4703 .br
4704 build: 2.1-DEV-rev155-g6bbb9e089-master
4705 .br
4706 Copyright: (c) 2000-2022 Telecom Paris distributed under LGPL v2.1+ - <http://gpac.io>
4707 .br
4708 .SH SEE ALSO
4709 .LP
4710 gpac-filters(1),MP4Box(1)

