

原 ffmpeg 源代码简单分析：avcodec_register_all()

2013年10月13日 13:34:01 阅读数：26512

=====

FFmpeg的库函数源代码分析文章列表：

【架构图】

[FFmpeg 源代码结构图 - 解码](#)

[FFmpeg 源代码结构图 - 编码](#)

【通用】

[FFmpeg 源代码简单分析：av_register_all\(\)](#)

[FFmpeg 源代码简单分析：avcodec_register_all\(\)](#)

[FFmpeg 源代码简单分析：内存的分配和释放（av_malloc\(\)、av_free\(\)等）](#)

[FFmpeg 源代码简单分析：常见结构体的初始化和销毁（AVFormatContext，AVFrame等）](#)

[FFmpeg 源代码简单分析：avio_open2\(\)](#)

[FFmpeg 源代码简单分析：av_find_decoder\(\)和av_find_encoder\(\)](#)

[FFmpeg 源代码简单分析：avcodec_open2\(\)](#)

[FFmpeg 源代码简单分析：avcodec_close\(\)](#)

【解码】

[图解FFMPEG 打开媒体的函数 avformat_open_input](#)

[FFmpeg 源代码简单分析：avformat_open_input\(\)](#)

[FFmpeg 源代码简单分析：avformat_find_stream_info\(\)](#)

[FFmpeg 源代码简单分析：av_read_frame\(\)](#)

[FFmpeg 源代码简单分析：avcodec_decode_video2\(\)](#)

[FFmpeg 源代码简单分析：avformat_close_input\(\)](#)

【编码】

[FFmpeg 源代码简单分析：avformat_alloc_output_context2\(\)](#)

[FFmpeg 源代码简单分析：avformat_write_header\(\)](#)

[FFmpeg 源代码简单分析：avcodec_encode_video\(\)](#)

[FFmpeg 源代码简单分析：av_write_frame\(\)](#)

[FFmpeg 源代码简单分析：av_write_trailer\(\)](#)

【其它】

[FFmpeg 源代码简单分析：日志输出系统（av_log\(\)等）](#)

[FFmpeg 源代码简单分析：结构体成员管理系统 -AVClass](#)

[FFmpeg 源代码简单分析：结构体成员管理系统 -AVOption](#)

[FFmpeg 源代码简单分析：libswscale 的 sws_getContext\(\)](#)

[FFmpeg 源代码简单分析：libswscale 的 sws_scale\(\)](#)

[FFmpeg 源代码简单分析：libavdevice 的 avdevice_register_all\(\)](#)

[FFmpeg 源代码简单分析：libavdevice 的 gdigrab](#)

【脚本】

[FFmpeg 源代码简单分析：makefile](#)

[FFmpeg 源代码简单分析：configure](#)

[【H.264】](#)

[FFmpeg 的 H.264 解码器源代码简单分析：概述](#)

=====

本文分析一下ffmpeg注册编解码器等函数avcodec_register_all()（注意不是av_register_all()，那是注册所有东西的）。该函数在所有基于ffmpeg的应用程序中几乎都是第一个被调用的。只有调用了该函数，才能使用编解码器等。

之前已经写过一篇文章了：

[ffmpeg 源代码简单分析：av_register_all\(\)](#)

其实 注册编解码器和注册复用器解复用器道理是差不多的，重复的内容不再多说。

编码器 的注册是：

```
REGISTER_ENCODER (X,x);
```

例如：

```
REGISTER_ENCODER (LJPEG, ljpeg);
```

解码器 的注册是：

```
REGISTER_DECODER (X,x);
```

例如：

```
REGISTER_DECODER (H264, h264);
```

既包含编码器有包含解码器 的注册是：

```
REGISTER_ENCDEC (X,x);
```

例如：

```
REGISTER_ENCDEC (BMP, bmp);
```

此外还有几种注册：

Parser：

```
REGISTER_PARSER (X,x);
```

例如：

```
REGISTER_PARSER (H264, h264);
```

BSF (bitstream filters, 比特流滤镜, 有一个常用：h264_mp4toannexb)：

```
REGISTER_BSF (X,x);
```

例如：

```
REGISTER_BSF (H264_MP4TOANNEXB, h264_mp4toannexb);
```

HWACCEL (hardware accelerators, 硬件加速器)：

```
REGISTER_HWACCEL (X,x);
```

例如：

```
REGISTER_HWACCEL (H264_DXVA2, h264_dxva2);
```

我们来看一下宏的定义，这里以 编解码器 为例：

```
[cpp]
1. #define REGISTER_ENCODER(X,x) { \
2.     extern AVCodec ff_###_encoder; \
3.     if(CONFIG_###_ENCODER) avcodec_register(&ff_###_encoder); }
4. #define REGISTER_DECODER(X,x) { \
5.     extern AVCodec ff_###_decoder; \
6.     if(CONFIG_###_DECODER) avcodec_register(&ff_###_decoder); }
7. #define REGISTER_ENCDEC(X,x) REGISTER_ENCODER(X,x); REGISTER_DECODER(X,x)
```

在这里，我发现其实编码器和解码器用的注册函数都是一样的：avcodec_register()

以REGISTER_DECODER (H264, h264)为例，就是等效于

```
[cpp]
1. extern AVCodec ff_h264_decoder;
2. if(CONFIG_H264_DECODER) avcodec_register(&ff_h264_decoder);
```

下面看一下 avcodec_register()的源代码：

```
[cpp]
1. //注册所有的AVCodec
2. void avcodec_register(AVCodec *codec)
3. {
4.     AVCodec **p;
5.     //初始化
6.     avcodec_init();
7.     //从第一个开始
8.     p = &first_avcodec;
9.     while (*p != NULL) p = &(*p)->next;
10.    *p = codec;
11.    codec->next = NULL;
12.
13.    if (codec->init_static_data)
14.        codec->init_static_data(codec);
15. }
```

这段代码是比较容易理解的。首先提一点，first_avcodec是就是AVCodec链表的头部地址，是一个全局静态变量，定义如下：

```
[cpp]
1. /* encoder management */
2. static AVCodec *first_avcodec = NULL;
```

由此我们可以分析出 avcodec_register() 的含义，一句话概括就是：遍历链表并把当前的AVCodec加到链表的尾部。

同理，Parser，BSF (bitstream filters, 比特流滤镜)，HWACCEL (hardware accelerators, 硬件加速器) 的注册方式都是类似的。不再详述。

下面贴出它的源代码：

```
[cpp]
1. /*
2.  *雷霄骅
3.  *leixiaohua1020@126.com
4.  *中国传媒大学/数字电视技术
5.  */
6. /*
7.  * Provide registration of all codecs, parsers and bitstream filters for libavcodec.
8.  * Copyright (c) 2002 Fabrice Bellard
9.  *
10. * This file is part of FFmpeg.
11. *
12. * FFmpeg is free software; you can redistribute it and/or
13. * modify it under the terms of the GNU Lesser General Public
14. * License as published by the Free Software Foundation; either
15. * version 2.1 of the License, or (at your option) any later version.
16. *
17. * FFmpeg is distributed in the hope that it will be useful,
18. * but WITHOUT ANY WARRANTY; without even the implied warranty of
19. * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU
20. * Lesser General Public License for more details.
21. *
22. * You should have received a copy of the GNU Lesser General Public
23. * License along with FFmpeg; if not, write to the Free Software
24. * Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
25. */
26.
27. /**
28.  * @file
29.  * Provide registration of all codecs, parsers and bitstream filters for libavcodec.
30.  */
31.
```

```

32. #include "avcodec.h"
33. //硬件加速
34. #define REGISTER_HWACCEL(X,x) { \
35.     extern AVHWAccel ff_##x##_hwaccel; \
36.     if(CONFIG_##X##_HWACCEL) av_register_hwaccel(&ff_##x##_hwaccel); }
37.
38. #define REGISTER_ENCODER(X,x) { \
39.     extern AVCodec ff_##x##_encoder; \
40.     if(CONFIG_##X##_ENCODER) avcodec_register(&ff_##x##_encoder); }
41. //定义的宏？宏的速度会快一点？注册AVCodec
42. //extern AVCodec ff_##x##_decoder;
43. //注意：extern表明全局唯一
44. //在h264中，对应的就是ff_h264_decoder
45. //由此可见AVCodecParser的名字是固定的
46. #define REGISTER_DECODER(X,x) { \
47.     extern AVCodec ff_##x##_decoder; \
48.     if(CONFIG_##X##_DECODER) avcodec_register(&ff_##x##_decoder); }
49. #define REGISTER_ENCDEC(X,x) REGISTER_ENCODER(X,x); REGISTER_DECODER(X,x)
50. //定义的宏？宏的速度会快一点？注册AVCodecParser
51. //extern AVCodecParser ff_##x##_parser;
52. //在h264中，对应的就是ff_h264_parser
53. //由此可见AVCodecParser的名字是固定的
54. #define REGISTER_PARSER(X,x) { \
55.     extern AVCodecParser ff_##x##_parser; \
56.     if(CONFIG_##X##_PARSER) av_register_codec_parser(&ff_##x##_parser); }
57. #define REGISTER_BSF(X,x) { \
58.     extern AVBitStreamFilter ff_##x##_bsf; \
59.     if(CONFIG_##X##_BSF) av_register_bitstream_filter(&ff_##x##_bsf); }
60.
61. void avcodec_register_all(void)
62. {
63.     static int initialized;
64.
65.     if (initialized)
66.         return;
67.     initialized = 1;
68.
69.     /* hardware accelerators */
70.     REGISTER_HWACCEL (H263_VAAPI, h263_vaapi);
71.     REGISTER_HWACCEL (H264_DXVA2, h264_dxva2);
72.     REGISTER_HWACCEL (H264_VAAPI, h264_vaapi);
73.     REGISTER_HWACCEL (H264_VDA, h264_vda);
74.     REGISTER_HWACCEL (MPEG1_VDPAU, mpeg1_vdpau);
75.     REGISTER_HWACCEL (MPEG2_DXVA2, mpeg2_dxva2);
76.     REGISTER_HWACCEL (MPEG2_VAAPI, mpeg2_vaapi);
77.     REGISTER_HWACCEL (MPEG2_VDPAU, mpeg2_vdpau);
78.     REGISTER_HWACCEL (MPEG4_VAAPI, mpeg4_vaapi);
79.     REGISTER_HWACCEL (VC1_DXVA2, vc1_dxva2);
80.     REGISTER_HWACCEL (VC1_VAAPI, vc1_vaapi);
81.     REGISTER_HWACCEL (WMV3_DXVA2, wmv3_dxva2);
82.     REGISTER_HWACCEL (WMV3_VAAPI, wmv3_vaapi);
83.
84.     /* video codecs */
85.     REGISTER_ENCODER (A64MULTI, a64multi);
86.     REGISTER_ENCODER (A64MULTI5, a64multi5);
87.     REGISTER_DECODER (AASC, aasc);
88.     REGISTER_ENCDEC (AMV, amv);
89.     REGISTER_DECODER (ANM, anm);
90.     REGISTER_DECODER (ANSI, ansi);
91.     REGISTER_ENCDEC (ASV1, asv1);
92.     REGISTER_ENCDEC (ASV2, asv2);
93.     REGISTER_DECODER (AURA, aura);
94.     REGISTER_DECODER (AURA2, aura2);
95.     REGISTER_ENCDEC (AVRP, avrp);
96.     REGISTER_DECODER (AVS, avs);
97.     REGISTER_DECODER (BETHSOFTVID, bethsoftvid);
98.     REGISTER_DECODER (BFI, bfi);
99.     REGISTER_DECODER (BINK, bink);
100.    REGISTER_ENCDEC (BMP, bmp);
101.    REGISTER_DECODER (BMV_VIDEO, bmv_video);
102.    REGISTER_DECODER (C93, c93);
103.    REGISTER_DECODER (CAVS, cavs);
104.    REGISTER_DECODER (CDGRAPHICS, cdgraphics);
105.    REGISTER_DECODER (CINEPAK, cinepak);
106.    REGISTER_ENCDEC (CLJR, cljr);
107.    REGISTER_DECODER (CSCD, cscd);
108.    REGISTER_DECODER (CYUV, cyuv);
109.    REGISTER_DECODER (DFA, dfa);
110.    REGISTER_DECODER (DIRAC, dirac);
111.    REGISTER_ENCDEC (DNXHD, dnxhd);
112.    REGISTER_ENCDEC (DPX, dpx);
113.    REGISTER_DECODER (DSICINVIDEO, dsicinvideo);
114.    REGISTER_ENCDEC (DVVIDEO, dvvideo);
115.    REGISTER_DECODER (DXA, dxa);
116.    REGISTER_DECODER (DXTORY, dxtory);
117.    REGISTER_DECODER (EACMV, eacmv);
118.    REGISTER_DECODER (EAMAD, eamad);
119.    REGISTER_DECODER (EATGQ, eatgq);
120.    REGISTER_DECODER (EATGV, eatgv);
121.    REGISTER_DECODER (EATQI, eatqi);
122.    REGISTER_DECODER (EIGHTBPS, eightbps);
123.    REGISTER_DECODER (EIGHTBPS, eightbps);

```

```

123. REGISTER_DECODER (EIGHTSVX_EXP, eightsvx_exp);
124. REGISTER_DECODER (EIGHTSVX_FIB, eightsvx_fib);
125. REGISTER_DECODER (ESCAPE124, escape124);
126. REGISTER_DECODER (ESCAPE130, escape130);
127. REGISTER_ENCDEC (FFV1, ffv1);
128. REGISTER_ENCDEC (FFVHUFF, ffvhuff);
129. REGISTER_ENCDEC (FLASHSV, flashsv);
130. REGISTER_ENCDEC (FLASHSV2, flashsv2);
131. REGISTER_DECODER (FLIC, flic);
132. REGISTER_ENCDEC (FLV, flv);
133. REGISTER_DECODER (FOURXM, fourxm);
134. REGISTER_DECODER (FRAPS, fraps);
135. REGISTER_DECODER (FRWU, frwu);
136. REGISTER_ENCDEC (GIF, gif);
137. REGISTER_ENCDEC (H261, h261);
138. REGISTER_ENCDEC (H263, h263);
139. REGISTER_DECODER (H263I, h263i);
140. REGISTER_ENCDEC (H263P, h263p);
141. REGISTER_DECODER (H264, h264);
142. REGISTER_DECODER (H264_CRYSTALHD, h264_crystalhd);
143. REGISTER_DECODER (H264_VDPAU, h264_vdpau);
144. REGISTER_ENCDEC (HUFFYUV, huffyuv);
145. REGISTER_DECODER (IDCIN, idcin);
146. REGISTER_DECODER (IFF_BYTERUN1, iff_byterun1);
147. REGISTER_DECODER (IFF_ILBM, iff_ilbm);
148. REGISTER_DECODER (INDE02, indeo2);
149. REGISTER_DECODER (INDE03, indeo3);
150. REGISTER_DECODER (INDE04, indeo4);
151. REGISTER_DECODER (INDE05, indeo5);
152. REGISTER_DECODER (INTERPLAY_VIDEO, interplay_video);
153. REGISTER_ENCDEC (JPEG2000, jpeg2000);
154. REGISTER_ENCDEC (JPEGLS, jpegls);
155. REGISTER_DECODER (JV, jv);
156. REGISTER_DECODER (KGV1, kgv1);
157. REGISTER_DECODER (KMVC, kmvc);
158. REGISTER_DECODER (LAGARITH, lagarith);
159. REGISTER_ENCDEC (LJPEG, ljpeg);
160. REGISTER_DECODER (LOCO, loco);
161. REGISTER_DECODER (MDEC, mdec);
162. REGISTER_DECODER (MIMIC, mimic);
163. REGISTER_ENCDEC (MJPEG, mjpeg);
164. REGISTER_DECODER (MJPEGB, mjpegb);
165. REGISTER_DECODER (MMVIDEO, mmvideo);
166. REGISTER_DECODER (MOTIONPIXELS, motionpixels);
167. REGISTER_DECODER (MPEG_XVMC, mpeg_xvmc);
168. REGISTER_ENCDEC (MPEG1VIDEO, mpeg1video);
169. REGISTER_ENCDEC (MPEG2VIDEO, mpeg2video);
170. REGISTER_ENCDEC (MPEG4, mpeg4);
171. REGISTER_DECODER (MPEG4_CRYSTALHD, mpeg4_crystalhd);
172. REGISTER_DECODER (MPEG4_VDPAU, mpeg4_vdpau);
173. REGISTER_DECODER (MPEGVIDEO, mpegvideo);
174. REGISTER_DECODER (MPEG_VDPAU, mpeg_vdpau);
175. REGISTER_DECODER (MPEG1_VDPAU, mpeg1_vdpau);
176. REGISTER_DECODER (MPEG2_CRYSTALHD, mpeg2_crystalhd);
177. REGISTER_DECODER (MSMPEG4_CRYSTALHD, msmpeg4_crystalhd);
178. REGISTER_DECODER (MSMPEG4V1, msmpeg4v1);
179. REGISTER_ENCDEC (MSMPEG4V2, msmpeg4v2);
180. REGISTER_ENCDEC (MSMPEG4V3, msmpeg4v3);
181. REGISTER_DECODER (MSRLE, msrle);
182. REGISTER_ENCDEC (MSVIDEO1, msvideo1);
183. REGISTER_DECODER (MSZH, mszh);
184. REGISTER_DECODER (MXPEG, mxpeg);
185. REGISTER_DECODER (NUV, nuv);
186. REGISTER_ENCDEC (PAM, pam);
187. REGISTER_ENCDEC (PBM, pbm);
188. REGISTER_ENCDEC (PCX, pcx);
189. REGISTER_ENCDEC (PGM, pgm);
190. REGISTER_ENCDEC (PGMYUV, pgmyuv);
191. REGISTER_DECODER (PICTOR, pictor);
192. REGISTER_ENCDEC (PNG, png);
193. REGISTER_ENCDEC (PPM, ppm);
194. REGISTER_ENCDEC (PRORES, prores);
195. REGISTER_DECODER (PRORES_LGPL, prores_lgpl);
196. REGISTER_DECODER (PTX, ptx);
197. REGISTER_DECODER (QDRAW, qdraw);
198. REGISTER_DECODER (QPEG, qpeg);
199. REGISTER_ENCDEC (QTRLE, qtrle);
200. REGISTER_ENCDEC (R10K, r10k);
201. REGISTER_ENCDEC (R210, r210);
202. REGISTER_ENCDEC (RAWVIDEO, rawvideo);
203. REGISTER_DECODER (RL2, rl2);
204. REGISTER_ENCDEC (ROQ, roq);
205. REGISTER_DECODER (RPZA, rpza);
206. REGISTER_ENCDEC (RV10, rv10);
207. REGISTER_ENCDEC (RV20, rv20);
208. REGISTER_DECODER (RV30, rv30);
209. REGISTER_DECODER (RV40, rv40);
210. REGISTER_DECODER (S302M, s302m);
211. REGISTER_ENCDEC (SGI, sgi);
212. REGISTER_DECODER (SMACKER, smacker);
213. REGISTER_DECODER (SMC, smc);
214. REGISTER_ENCDEC (SMOW, smow);

```

```

214. REGISTER_ENCODER (SHOW, show);
215. REGISTER_DECODER (SP5X, sp5x);
216. REGISTER_DECODER (SUNRAST, sunrast);
217. REGISTER_ENCODER (SVQ1, svq1);
218. REGISTER_DECODER (SVQ3, svq3);
219. REGISTER_ENCODER (TARGA, targa);
220. REGISTER_DECODER (THEORA, theora);
221. REGISTER_DECODER (THP, thp);
222. REGISTER_DECODER (TIERTEXSEQVIDEO, tiertexseqvideo);
223. REGISTER_ENCODER (TIFF, tiff);
224. REGISTER_DECODER (TMV, tmv);
225. REGISTER_DECODER (TRUEMOTION1, truemotion1);
226. REGISTER_DECODER (TRUEMOTION2, truemotion2);
227. REGISTER_DECODER (TSCC, tsc);
228. REGISTER_DECODER (TXD, txd);
229. REGISTER_DECODER (ULTI, ulti);
230. REGISTER_DECODER (UTVIDEO, utvideo);
231. REGISTER_ENCODER (V210, v210);
232. REGISTER_DECODER (V210X, v210x);
233. REGISTER_ENCODER (V308, v308);
234. REGISTER_ENCODER (V410, v410);
235. REGISTER_DECODER (VB, vb);
236. REGISTER_DECODER (VBLE, vble);
237. REGISTER_DECODER (VC1, vc1);
238. REGISTER_DECODER (VC1_CRYSTALHD, vc1_crystalhd);
239. REGISTER_DECODER (VC1_VDPAU, vc1_vdpau);
240. REGISTER_DECODER (VC1IMAGE, vc1image);
241. REGISTER_DECODER (VCR1, vcr1);
242. REGISTER_DECODER (VMDVIDEO, vmdvideo);
243. REGISTER_DECODER (VMNC, vmnc);
244. REGISTER_DECODER (VP3, vp3);
245. REGISTER_DECODER (VP5, vp5);
246. REGISTER_DECODER (VP6, vp6);
247. REGISTER_DECODER (VP6A, vp6a);
248. REGISTER_DECODER (VP6F, vp6f);
249. REGISTER_DECODER (VP8, vp8);
250. REGISTER_DECODER (VQA, vqa);
251. REGISTER_ENCODER (WMV1, wmv1);
252. REGISTER_ENCODER (WMV2, wmv2);
253. REGISTER_DECODER (WMV3, wmv3);
254. REGISTER_DECODER (WMV3_CRYSTALHD, wmv3_crystalhd);
255. REGISTER_DECODER (WMV3_VDPAU, wmv3_vdpau);
256. REGISTER_DECODER (WMV3IMAGE, wmv3image);
257. REGISTER_DECODER (WNV1, wnv1);
258. REGISTER_DECODER (XAN_WC3, xan_wc3);
259. REGISTER_DECODER (XAN_WC4, xan_wc4);
260. REGISTER_DECODER (XL, xl);
261. REGISTER_ENCODER (XWD, xwd);
262. REGISTER_ENCODER (Y41P, y41p);
263. REGISTER_DECODER (YOP, yop);
264. REGISTER_ENCODER (YUV4, yuv4);
265. REGISTER_ENCODER (ZLIB, zlib);
266. REGISTER_ENCODER (ZMBV, zmbv);
267.
268. /* audio codecs */
269. REGISTER_ENCODER (AAC, aac);
270. REGISTER_DECODER (AAC_LATM, aac_latm);
271. REGISTER_ENCODER (AC3, ac3);
272. REGISTER_ENCODER (AC3_FIXED, ac3_fixed);
273. REGISTER_ENCODER (ALAC, alac);
274. REGISTER_DECODER (ALS, als);
275. REGISTER_DECODER (AMRNB, amrnb);
276. REGISTER_DECODER (AMRWB, amrwb);
277. REGISTER_DECODER (APE, ape);
278. REGISTER_DECODER (ATRAC1, atrac1);
279. REGISTER_DECODER (ATRAC3, atrac3);
280. REGISTER_DECODER (BINKAUDIO_DCT, binkaudio_dct);
281. REGISTER_DECODER (BINKAUDIO_RDFT, binkaudio_rdft);
282. REGISTER_DECODER (BMV_AUDIO, bmv_audio);
283. REGISTER_DECODER (COOK, cook);
284. REGISTER_ENCODER (DCA, dca);
285. REGISTER_DECODER (DSICINAUDIO, dsicinaudio);
286. REGISTER_ENCODER (EAC3, eac3);
287. REGISTER_DECODER (FFWAVESYNTH, ffwavesynth);
288. REGISTER_ENCODER (FLAC, flac);
289. REGISTER_ENCODER (G723_1, g723_1);
290. REGISTER_DECODER (G729, g729);
291. REGISTER_DECODER (GSM, gsm);
292. REGISTER_DECODER (GSM_MS, gsm_ms);
293. REGISTER_DECODER (IMC, imc);
294. REGISTER_DECODER (MACE3, mace3);
295. REGISTER_DECODER (MACE6, mace6);
296. REGISTER_DECODER (MLP, mlp);
297. REGISTER_DECODER (MP1, mp1);
298. REGISTER_DECODER (MP1FLOAT, mp1float);
299. REGISTER_ENCODER (MP2, mp2);
300. REGISTER_DECODER (MP2FLOAT, mp2float);
301. REGISTER_DECODER (MP3, mp3);
302. REGISTER_DECODER (MP3FLOAT, mp3float);
303. REGISTER_DECODER (MP3ADU, mp3adu);
304. REGISTER_DECODER (MP3ADUFLOAT, mp3adufloat);
305. REGISTER_DECODER (MP3ON4, mp3on4);

```

```

306. REGISTER_DECODER (MP3ON4FLOAT, mp3on4float);
307. REGISTER_DECODER (MPC7, mpc7);
308. REGISTER_DECODER (MPC8, mpc8);
309. REGISTER_ENCDEC (NELLYMOSER, nellymoser);
310. REGISTER_DECODER (QCELP, qcelp);
311. REGISTER_DECODER (QDM2, qdm2);
312. REGISTER_ENCDEC (RA_144, ra_144);
313. REGISTER_DECODER (RA_288, ra_288);
314. REGISTER_DECODER (SHORTEN, shorten);
315. REGISTER_DECODER (SIPR, sipr);
316. REGISTER_DECODER (SMACKAUD, smackaud);
317. REGISTER_ENCDEC (SONIC, sonic);
318. REGISTER_ENCODER (SONIC_LS, sonic_ls);
319. REGISTER_DECODER (TRUEHD, truehd);
320. REGISTER_DECODER (TRUESPEECH, truespeech);
321. REGISTER_DECODER (TTA, tta);
322. REGISTER_DECODER (TWINVQ, twinvq);
323. REGISTER_DECODER (VMDAUDIO0, vmdaudio);
324. REGISTER_ENCDEC (VORBIS, vorbis);
325. REGISTER_DECODER (WAVPACK, wavpack);
326. REGISTER_DECODER (WMALOSSLESS, wmalossless);
327. REGISTER_DECODER (WMAPRO, wmapro);
328. REGISTER_ENCDEC (WMAV1, wma1);
329. REGISTER_ENCDEC (WMAV2, wma2);
330. REGISTER_DECODER (WMAVOICE, wmavoice);
331. REGISTER_DECODER (WS_SND1, ws_snd1);
332.
333. /* PCM codecs */
334. REGISTER_ENCDEC (PCM_ALAW, pcm_alaw);
335. REGISTER_DECODER (PCM_BLURAY, pcm_bluray);
336. REGISTER_DECODER (PCM_DVD, pcm_dvd);
337. REGISTER_ENCDEC (PCM_F32BE, pcm_f32be);
338. REGISTER_ENCDEC (PCM_F32LE, pcm_f32le);
339. REGISTER_ENCDEC (PCM_F64BE, pcm_f64be);
340. REGISTER_ENCDEC (PCM_F64LE, pcm_f64le);
341. REGISTER_DECODER (PCM_LXF, pcm_lxf);
342. REGISTER_ENCDEC (PCM_MULAW, pcm_mulaw);
343. REGISTER_ENCDEC (PCM_S8, pcm_s8);
344. REGISTER_DECODER (PCM_S8_PLANAR, pcm_s8_planar);
345. REGISTER_ENCDEC (PCM_S16BE, pcm_s16be);
346. REGISTER_ENCDEC (PCM_S16LE, pcm_s16le);
347. REGISTER_DECODER (PCM_S16LE_PLANAR, pcm_s16le_planar);
348. REGISTER_ENCDEC (PCM_S24BE, pcm_s24be);
349. REGISTER_ENCDEC (PCM_S24DAUD, pcm_s24daud);
350. REGISTER_ENCDEC (PCM_S24LE, pcm_s24le);
351. REGISTER_ENCDEC (PCM_S32BE, pcm_s32be);
352. REGISTER_ENCDEC (PCM_S32LE, pcm_s32le);
353. REGISTER_ENCDEC (PCM_U8, pcm_u8);
354. REGISTER_ENCDEC (PCM_U16BE, pcm_u16be);
355. REGISTER_ENCDEC (PCM_U16LE, pcm_u16le);
356. REGISTER_ENCDEC (PCM_U24BE, pcm_u24be);
357. REGISTER_ENCDEC (PCM_U24LE, pcm_u24le);
358. REGISTER_ENCDEC (PCM_U32BE, pcm_u32be);
359. REGISTER_ENCDEC (PCM_U32LE, pcm_u32le);
360. REGISTER_DECODER (PCM_ZORK, pcm_zork);
361.
362. /* DPCM codecs */
363. REGISTER_DECODER (INTERPLAY_DPCM, interplay_dpcm);
364. REGISTER_ENCDEC (ROQ_DPCM, roq_dpcm);
365. REGISTER_DECODER (SOL_DPCM, sol_dpcm);
366. REGISTER_DECODER (XAN_DPCM, xan_dpcm);
367.
368. /* ADPCM codecs */
369. REGISTER_DECODER (ADPCM_4XM, adpcm_4xm);
370. REGISTER_ENCDEC (ADPCM_ADX, adpcm_adx);
371. REGISTER_DECODER (ADPCM_CT, adpcm_ct);
372. REGISTER_DECODER (ADPCM_EA, adpcm_ea);
373. REGISTER_DECODER (ADPCM_EA_MAXIS_XA, adpcm_ea_maxis_xa);
374. REGISTER_DECODER (ADPCM_EA_R1, adpcm_ea_r1);
375. REGISTER_DECODER (ADPCM_EA_R2, adpcm_ea_r2);
376. REGISTER_DECODER (ADPCM_EA_R3, adpcm_ea_r3);
377. REGISTER_DECODER (ADPCM_EA_XAS, adpcm_ea_xas);
378. REGISTER_ENCDEC (ADPCM_G722, adpcm_g722);
379. REGISTER_ENCDEC (ADPCM_G726, adpcm_g726);
380. REGISTER_DECODER (ADPCM_IMA_AMV, adpcm_ima_amv);
381. REGISTER_DECODER (ADPCM_IMA_APC, adpcm_ima_apc);
382. REGISTER_DECODER (ADPCM_IMA_DK3, adpcm_ima_dk3);
383. REGISTER_DECODER (ADPCM_IMA_DK4, adpcm_ima_dk4);
384. REGISTER_DECODER (ADPCM_IMA_EA_EACS, adpcm_ima_ea_eacs);
385. REGISTER_DECODER (ADPCM_IMA_EA_SEAD, adpcm_ima_ea_sead);
386. REGISTER_DECODER (ADPCM_IMA_ISS, adpcm_ima_iss);
387. REGISTER_ENCDEC (ADPCM_IMA_QT, adpcm_ima_qt);
388. REGISTER_DECODER (ADPCM_IMA_SMJPEG, adpcm_ima_smjpeg);
389. REGISTER_ENCDEC (ADPCM_IMA_WAV, adpcm_ima_wav);
390. REGISTER_DECODER (ADPCM_IMA_WS, adpcm_ima_ws);
391. REGISTER_ENCDEC (ADPCM_MS, adpcm_ms);
392. REGISTER_DECODER (ADPCM_SBPRO_2, adpcm_sbpro_2);
393. REGISTER_DECODER (ADPCM_SBPRO_3, adpcm_sbpro_3);
394. REGISTER_DECODER (ADPCM_SBPRO_4, adpcm_sbpro_4);
395. REGISTER_ENCDEC (ADPCM_SWF, adpcm_swf);
396. REGISTER_DECODER (ADPCM_THP, adpcm_thp);

```

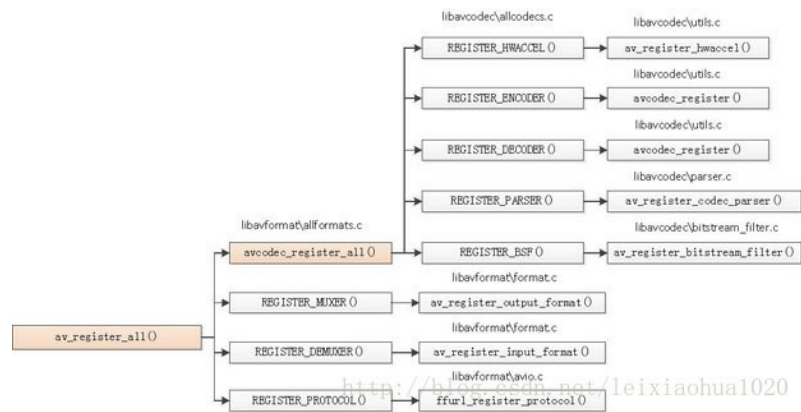
```

397. REGISTER_DECODER (ADPCM_XA, adpcm_xa);
398. REGISTER_ENCDEC (ADPCM_YAMAHA, adpcm_yamaha);
399.
400. /* subtitles */
401. REGISTER_ENCDEC (ASS, ass);
402. REGISTER_ENCDEC (DVBSUB, dvbsub);
403. REGISTER_ENCDEC (DVDSUB, dvdsup);
404. REGISTER_DECODER (PGSSUB, pgssub);
405. REGISTER_ENCDEC (SRT, srt);
406. REGISTER_ENCDEC (XSUB, xsub);
407.
408. /* external libraries */
409. REGISTER_ENCODER (LIBAACPLUS, libaacplus);
410. REGISTER_DECODER (LIBCELT, libcelt);
411. REGISTER_ENCDEC (LIBDIRAC, libdirac);
412. REGISTER_ENCODER (LIBFAAC, libfaac);
413. REGISTER_ENCDEC (LIBGSM, libgsm);
414. REGISTER_ENCDEC (LIBGSM_MS, libgsm_ms);
415. REGISTER_ENCODER (LIBMP3LAME, libmp3lame);
416. REGISTER_ENCDEC (LIBOPENCORE_AMRNB, libopencore_amrnb);
417. REGISTER_DECODER (LIBOPENCORE_AMRWB, libopencore_amrwb);
418. REGISTER_ENCDEC (LIBOPENJPEG, libopenjpeg);
419. REGISTER_ENCDEC (LIBSCHROEDINGER, libschroedinger);
420. REGISTER_ENCDEC (LIBSPEEX, libspeex);
421. REGISTER_DECODER (LIBSTAGEFRIGHT_H264, libstagefright_h264);
422. REGISTER_ENCODER (LIBTHEORA, libtheora);
423. REGISTER_DECODER (LIBUTVIDEO, libutvideo);
424. REGISTER_ENCODER (LIBVO_AACENC, libvo_aacenc);
425. REGISTER_ENCODER (LIBVO_AMRWBENC, libvo_amrwbenc);
426. REGISTER_ENCODER (LIBVORBIS, libvorbis);
427. REGISTER_ENCDEC (LIBVPX, libvpx);
428. REGISTER_ENCODER (LIBX264, libx264);
429. REGISTER_ENCODER (LIBX264RGB, libx264rgb);
430. REGISTER_ENCODER (LIBXAVS, libxavs);
431. REGISTER_ENCODER (LIBXVID, libxvid);
432.
433. /* text */
434. REGISTER_DECODER (BINTEXT, bintext);
435. REGISTER_DECODER (XBIN, xbin);
436. REGISTER_DECODER (IDF, idf);
437.
438. /* parsers */
439. REGISTER_PARSER (AAC, aac);
440. REGISTER_PARSER (AAC_LATM, aac_latm);
441. REGISTER_PARSER (AC3, ac3);
442. REGISTER_PARSER (ADX, adx);
443. REGISTER_PARSER (CAVSDVIDEO, cavsvideo);
444. REGISTER_PARSER (DCA, dca);
445. REGISTER_PARSER (DIRAC, dirac);
446. REGISTER_PARSER (DNXHD, dnxhd);
447. REGISTER_PARSER (DVBSUB, dvbsub);
448. REGISTER_PARSER (DVDSUB, dvdsup);
449. REGISTER_PARSER (FLAC, flac);
450. REGISTER_PARSER (GSM, gsm);
451. REGISTER_PARSER (H261, h261);
452. REGISTER_PARSER (H263, h263);
453. REGISTER_PARSER (H264, h264);
454. REGISTER_PARSER (MJPEG, mjpeg);
455. REGISTER_PARSER (MLP, mlp);
456. REGISTER_PARSER (MPEG4VIDEO, mpeg4video);
457. REGISTER_PARSER (MPEGAUDIO, mpegaudio);
458. REGISTER_PARSER (MPEGVIDEO, mpegvideo);
459. REGISTER_PARSER (PNM, pnm);
460. REGISTER_PARSER (RV30, rv30);
461. REGISTER_PARSER (RV40, rv40);
462. REGISTER_PARSER (VC1, vc1);
463. REGISTER_PARSER (VP3, vp3);
464. REGISTER_PARSER (VP8, vp8);
465.
466. /* bitstream filters */
467. REGISTER_BSF (AAC_ADTSTOASC, aac_adtstoasc);
468. REGISTER_BSF (CHOMP, chomp);
469. REGISTER_BSF (DUMP_EXTRADATA, dump_extradata);
470. REGISTER_BSF (H264_MP4TOANNEXB, h264_mp4toannexb);
471. REGISTER_BSF (IMX_DUMP_HEADER, imx_dump_header);
472. REGISTER_BSF (MJPEG2JPEG, mjpeg2jpeg);
473. REGISTER_BSF (MJPEGA_DUMP_HEADER, mjpega_dump_header);
474. REGISTER_BSF (MP3_HEADER_COMPRESS, mp3_header_compress);
475. REGISTER_BSF (MP3_HEADER_DECOMPRESS, mp3_header_decompress);
476. REGISTER_BSF (MOV2TEXTSUB, mov2textsub);
477. REGISTER_BSF (NOISE, noise);
478. REGISTER_BSF (REMOVE_EXTRADATA, remove_extradata);
479. REGISTER_BSF (TEXT2MOVSUB, text2movsub);
480. }

```

整个代码的过程就是首先确定是不是已经初始化过了（initialized），如果没有，就注册，注册，注册...直到完成所有注册。

函数的调用关系图如下图所示。av_register_all()调用了avcodec_register_all()。因此如果调用过av_register_all()的话就不需要再调用avcodec_register_all()了。



下面附上硬件加速器，编码器/解码器，parser，Bitstream Filter的注册代码。

硬件加速器注册函数是av_register_hwaccel()。

```
[cpp]
1. void av_register_hwaccel(AVHWAccel *hwaccel)
2. {
3.     AVHWAccel **p = last_hwaccel;
4.     hwaccel->next = NULL;
5.     while(*p || avpriv_atomic_ptr_cas((void * volatile *)p, NULL, hwaccel))
6.         p = &(*p)->next;
7.     last_hwaccel = &hwaccel->next;
8. }
```

编解码器注册函数是avcodec_register()。

```
[cpp]
1. av_cold void avcodec_register(AVCodec *codec)
2. {
3.     AVCodec **p;
4.     avcodec_init();
5.     p = last_avcodec;
6.     codec->next = NULL;
7.
8.     while(*p || avpriv_atomic_ptr_cas((void * volatile *)p, NULL, codec))
9.         p = &(*p)->next;
10.    last_avcodec = &codec->next;
11.
12.    if (codec->init_static_data)
13.        codec->init_static_data(codec);
14. }
```

parser注册函数是av_register_codec_parser()。

```
[cpp]
1. void av_register_codec_parser(AVCodecParser *parser)
2. {
3.     do {
4.         parser->next = av_first_parser;
5.     } while (parser->next != avpriv_atomic_ptr_cas((void * volatile *)&av_first_parser, parser->next, parser));
6. }
```

Bitstream Filter注册函数是av_register_bitstream_filter()。

```
[cpp]
1. void av_register_bitstream_filter(AVBitStreamFilter *bsf)
2. {
3.     do {
4.         bsf->next = first_bitstream_filter;
5.     } while(bsf->next != avpriv_atomic_ptr_cas((void * volatile *)&first_bitstream_filter, bsf->next, bsf));
6. }
```

后两个函数中的avpriv_atomic_ptr_cas()定义如下。

```
[cpp]
1. void *avpriv_atomic_ptr_cas(void * volatile *ptr, void *oldval, void *newval)
2. {
3.     if (*ptr == oldval) {
4.         *ptr = newval;
5.         return oldval;
6.     }
7.     return *ptr;
8. }
```

版权声明：本文为博主原创文章，未经博主允许不得转载。 <https://blog.csdn.net/leixiaohua1020/article/details/12677265>

文章标签：[ffmpeg](#) [源代码](#) [avcodec_register_all](#) [编码器](#) [解码器](#)

个人分类：[FFMPEG](#)

所属专栏：[开源多媒体项目源代码分析](#) [FFmpeg](#)

此PDF由[spygg](#)生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com