

=====

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](#)

=====

前一阵子看了一下ffmpeg的源代码，并且做了一些注释，在此贴出来以作备忘。

本文分析一下ffmpeg注册复用器，编码器等的函数av_register_all()。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，才能使用复用器，编码器。

可见 **解复用器** 注册都是用

```
REGISTER_DEMUXER (X,x)
```

例如：

```
REGISTER_DEMUXER (AAC, aac)
```

可见 **复用器** 注册都是用

```
REGISTER_MUXER (X,x)
```

例如：

```
REGISTER_MUXER (ADTS, adts)
```

既有解复用器又有复用器 的话，可以用

```
REGISTER_MUXDEMUX (X,x);
```

例如：

```
REGISTER_MUXDEMUX (AC3, ac3);
```

我们来看一下宏的定义，这里以 **解复用器** 为例：

```
[cpp]
1. #define REGISTER_DEMUXER(X,x) { \
2.     extern AVInputFormat ff_###_demuxer; \
3.     if(CONFIG_###_DEMUXER) av_register_input_format(&ff_###_demuxer); }
```

注意：define里面的##可能不太常见，它的含义就是拼接两个字符串，比如

```
#define Conn(x,y) x##y
```

那么

```
int n = Conn(123,456); 结果就是n=123456;
```

我们以REGISTER_DEMUXER (AAC, aac)为例，则它等效于

```
[cpp]
1. extern AVInputFormat ff_aac_demuxer;
2. if(CONFIG_AAC_DEMUXER) av_register_input_format(&ff_aac_demuxer);
```

从上面这段代码我们可以看出，真正注册的函数是av_register_input_format(&ff_aac_demuxer)，那我就看看这个和函数的作用，查看一下av_register_input_format()的代码：

```
[cpp]
1. void av_register_input_format(AVInputFormat *format)
2. {
3.     AVInputFormat **p;
4.     p = &first_iformat;
5.     while (*p != NULL) p = &(*p)->next;
6.     *p = format;
7.     format->next = NULL;
8. }
```

这段代码是比较容易理解的，首先提一点，first_iformat是个什么东东呢？其实它是Input Format链表的头部地址，是一个全局静态变量，定义如下：

```
[cpp]
1. /** head of registered input format linked list */
2. static AVInputFormat *first_iformat = NULL;
```

由此我们可以分析出av_register_input_format()的含义，一句话概括就是：遍历链表并把当前的Input Format加到链表的尾部。至此REGISTER_DEMUXER (X, x)分析完毕。

同理，复用器道理是一样的，只是注册函数改为av_register_output_format()；

既有解复用器又有复用器的话，有一个宏定义：

```
[cpp]
1. #define REGISTER_MUXDEMUX(X,x) REGISTER_MUXER(X,x); REGISTER_DEMUXER(X,x)
```

可见是分别注册了复用器和解复用器。

此外还有网络协议的注册，注册函数为ffurl_register_protocol()，在此不再详述。

下面贴出它的源代码（allformats.c）

```
[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.  * Register all the formats and protocols
8.  * Copyright (c) 2000, 2001, 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. #include "avformat.h"
27. #include "rtp.h"
28. #include "rdt.h"
29. #include "url.h"
30. //定义的宏？宏的速度会快一点？注册AVOutputFormat
31. //define中，用来把参数转换成字符串，##则用来连接前后两个参数，把它们变成一个字符串。
32. //感觉有点像Java中的EL，可以随意拼接字符串
33. #define REGISTER_MUXER(X,x) { \
34.     extern AVOutputFormat ff_##x##_muxer; \
35.     if(CONFIG_##X##_MUXER) av_register_output_format(&ff_##x##_muxer); }
36. //定义的宏？宏的速度会快一点？注册AVInputFormat
37. #define REGISTER_DEMUXER(X,x) { \
38.     extern AVInputFormat ff_##x##_demuxer; \
39.     if(CONFIG_##X##_DEMUXER) av_register_input_format(&ff_##x##_demuxer); }
40. //注册函数av_register_input_format
41.
42. //定义的宏？宏的速度会快一点？两个一起注册！
43. #define REGISTER_MUXDEMUX(X,x) REGISTER_MUXER(X,x); REGISTER_DEMUXER(X,x)
44. //定义的宏？宏的速度会快一点？注册URLProtocol
45. //extern URLProtocol ff_##x##_protocol;
46. //在librtmp中，对应的就是ff_rtmp_protocol
```

```

47. //这样就把librtmp整合起来了
48. //由此可见URLProtocol的名字是固定的
49. #define REGISTER_PROTOCOL(X,x) { \
50.     extern URLProtocol ff_###_protocol; \
51.     if(CONFIG_###_PROTOCOL) ffurl_register_protocol(&ff_###_protocol, sizeof(ff_###_protocol)); }
52. //注册函数ffurl_register_protocol
53. void av_register_all(void)
54. {
55.     static int initialized;
56.
57.     if (initialized)
58.         return;
59.     initialized = 1;
60.     //注册所有的codec
61.     avcodec_register_all();
62.     //注册所有的MUXER (复用器和解复用器)
63.     /* (de)muxers */
64.     REGISTER_MUXER    (A64, a64);
65.     REGISTER_DEMUXER  (AAC, aac);
66.     REGISTER_MUXDEMUX (AC3, ac3);
67.     REGISTER_DEMUXER  (ACT, act);
68.     REGISTER_DEMUXER  (ADF, adf);
69.     REGISTER_MUXER    (ADTS, adts);
70.     REGISTER_MUXDEMUX (ADX, adx);
71.     REGISTER_DEMUXER  (AEA, aea);
72.     REGISTER_MUXDEMUX (AIFF, aiff);
73.     REGISTER_MUXDEMUX (AMR, amr);
74.     REGISTER_DEMUXER  (ANM, anm);
75.     REGISTER_DEMUXER  (APC, apc);
76.     REGISTER_DEMUXER  (APE, ape);
77.     REGISTER_DEMUXER  (APPLEHTTP, applehttp);
78.     REGISTER_MUXDEMUX (ASF, asf);
79.     REGISTER_MUXDEMUX (ASS, ass);
80.     REGISTER_MUXER    (ASF_STREAM, asf_stream);
81.     REGISTER_MUXDEMUX (AU, au);
82.     REGISTER_MUXDEMUX (AVI, avi);
83.     REGISTER_DEMUXER  (AVISYNTH, avisynth);
84.     REGISTER_MUXER    (AVM2, avm2);
85.     REGISTER_DEMUXER  (AVS, avs);
86.     REGISTER_DEMUXER  (BETHSOFTVID, bethsoftvid);
87.     REGISTER_DEMUXER  (BFI, bfi);
88.     REGISTER_DEMUXER  (BINTEXT, bintext);
89.     REGISTER_DEMUXER  (BINK, bink);
90.     REGISTER_MUXDEMUX (BIT, bit);
91.     REGISTER_DEMUXER  (BMV, bmv);
92.     REGISTER_DEMUXER  (C93, c93);
93.     REGISTER_MUXDEMUX (CAF, caf);
94.     REGISTER_MUXDEMUX (CAVSVIDEO, cavsvideo);
95.     REGISTER_DEMUXER  (CDG, cdg);
96.     REGISTER_MUXER    (CRC, crc);
97.     REGISTER_MUXDEMUX (DAUD, daud);
98.     REGISTER_DEMUXER  (DFA, dfa);
99.     REGISTER_MUXDEMUX (DIRAC, dirac);
100.    REGISTER_MUXDEMUX (DNXHD, dnxhd);
101.    REGISTER_DEMUXER  (DSICIN, dsicin);
102.    REGISTER_MUXDEMUX (DTS, dts);
103.    REGISTER_MUXDEMUX (DV, dv);
104.    REGISTER_DEMUXER  (DXA, dxa);
105.    REGISTER_DEMUXER  (EA, ea);
106.    REGISTER_DEMUXER  (EA_CDATA, ea_cdata);
107.    REGISTER_MUXDEMUX (EAC3, eac3);
108.    REGISTER_MUXDEMUX (FFM, ffm);
109.    REGISTER_MUXDEMUX (FFMETADATA, ffmetadata);
110.    REGISTER_MUXDEMUX (FILMSTRIP, filmstrip);
111.    REGISTER_MUXDEMUX (FLAC, flac);
112.    REGISTER_DEMUXER  (FLIC, flic);
113.    REGISTER_MUXDEMUX (FLV, flv);
114.    REGISTER_DEMUXER  (FOURXM, fourxm);
115.    REGISTER_MUXER    (FRAMECRC, framecrc);
116.    REGISTER_MUXER    (FRAMEMD5, framemd5);
117.    REGISTER_MUXDEMUX (G722, g722);
118.    REGISTER_MUXDEMUX (G723_1, g723_1);
119.    REGISTER_DEMUXER  (G729, g729);
120.    REGISTER_MUXER    (GIF, gif);
121.    REGISTER_DEMUXER  (GSM, gsm);
122.    REGISTER_MUXDEMUX (GXF, gxf);
123.    REGISTER_MUXDEMUX (H261, h261);
124.    REGISTER_MUXDEMUX (H263, h263);
125.    REGISTER_MUXDEMUX (H264, h264);
126.    REGISTER_DEMUXER  (ICO, ico);
127.    REGISTER_DEMUXER  (IDCIN, idcin);
128.    REGISTER_DEMUXER  (IDF, idf);
129.    REGISTER_DEMUXER  (IFF, iff);
130.    REGISTER_MUXDEMUX (IMAGE2, image2);
131.    REGISTER_MUXDEMUX (IMAGE2PIPE, image2pipe);
132.    REGISTER_DEMUXER  (INGENIENT, ingenient);
133.    REGISTER_DEMUXER  (IPMOVIE, ipmovie);
134.    REGISTER_MUXER    (IPOD, ipod);
135.    REGISTER_MUXER    (ISMV, ismv);
136.    REGISTER_DEMUXER  (ISS, iss);
137.    REGISTER_DEMUXER  (IV8, iv8);

```

```

138. REGISTER_MUXDEMUX (IVF, ivf);
139. REGISTER_DEMUXER (JV, jv);
140. REGISTER_MUXDEMUX (LATM, latm);
141. REGISTER_DEMUXER (LMLM4, lmlm4);
142. REGISTER_DEMUXER (LOAS, loas);
143. REGISTER_DEMUXER (LXF, lxf);
144. REGISTER_MUXDEMUX (M4V, m4v);
145. REGISTER_MUXER (MDS, md5);
146. REGISTER_MUXDEMUX (MATROSKA, matroska);
147. REGISTER_MUXER (MATROSKA_AUDIO, matroska_audio);
148. REGISTER_MUXDEMUX (MICRODVD, microdvd);
149. REGISTER_MUXDEMUX (MJPEG, mjpeg);
150. REGISTER_MUXDEMUX (MLP, mlp);
151. REGISTER_DEMUXER (MM, mm);
152. REGISTER_MUXDEMUX (MMF, mmf);
153. REGISTER_MUXDEMUX (MOV, mov);
154. REGISTER_MUXER (MP2, mp2);
155. REGISTER_MUXDEMUX (MP3, mp3);
156. REGISTER_MUXER (MP4, mp4);
157. REGISTER_DEMUXER (MPC, mpc);
158. REGISTER_DEMUXER (MPC8, mpc8);
159. REGISTER_MUXER (MPEG1SYSTEM, mpeg1system);
160. REGISTER_MUXER (MPEG1VCD, mpeg1vcd);
161. REGISTER_MUXER (MPEG1VIDEO, mpeg1video);
162. REGISTER_MUXER (MPEG2DVD, mpeg2dvd);
163. REGISTER_MUXER (MPEG2SVCD, mpeg2svcd);
164. REGISTER_MUXER (MPEG2VIDEO, mpeg2video);
165. REGISTER_MUXER (MPEG2VOB, mpeg2vob);
166. REGISTER_DEMUXER (MPEGPS, mpegps);
167. REGISTER_MUXDEMUX (MPEGTS, mpegts);
168. REGISTER_DEMUXER (MPEGTSRAW, mpegtsraw);
169. REGISTER_DEMUXER (MPEGVIDEO, mpegvideo);
170. REGISTER_MUXER (MPJPEG, mpjpeg);
171. REGISTER_DEMUXER (MSNWC_TCP, msnwc_tcp);
172. REGISTER_DEMUXER (MTV, mtv);
173. REGISTER_DEMUXER (MVI, mvi);
174. REGISTER_MUXDEMUX (MXF, mxf);
175. REGISTER_MUXER (MXF_D10, mxf_d10);
176. REGISTER_DEMUXER (MXG, mxg);
177. REGISTER_DEMUXER (NC, nc);
178. REGISTER_DEMUXER (NSV, nsv);
179. REGISTER_MUXER (NULL, null);
180. REGISTER_MUXDEMUX (NUT, nut);
181. REGISTER_DEMUXER (NUV, nuv);
182. REGISTER_MUXDEMUX (OGG, ogg);
183. REGISTER_MUXDEMUX (OMA, oma);
184. REGISTER_MUXDEMUX (PCM_ALAW, pcm_alaw);
185. REGISTER_MUXDEMUX (PCM_MULAW, pcm_mulaw);
186. REGISTER_MUXDEMUX (PCM_F64BE, pcm_f64be);
187. REGISTER_MUXDEMUX (PCM_F64LE, pcm_f64le);
188. REGISTER_MUXDEMUX (PCM_F32BE, pcm_f32be);
189. REGISTER_MUXDEMUX (PCM_F32LE, pcm_f32le);
190. REGISTER_MUXDEMUX (PCM_S32BE, pcm_s32be);
191. REGISTER_MUXDEMUX (PCM_S32LE, pcm_s32le);
192. REGISTER_MUXDEMUX (PCM_S24BE, pcm_s24be);
193. REGISTER_MUXDEMUX (PCM_S24LE, pcm_s24le);
194. REGISTER_MUXDEMUX (PCM_S16BE, pcm_s16be);
195. REGISTER_MUXDEMUX (PCM_S16LE, pcm_s16le);
196. REGISTER_MUXDEMUX (PCM_S8, pcm_s8);
197. REGISTER_MUXDEMUX (PCM_U32BE, pcm_u32be);
198. REGISTER_MUXDEMUX (PCM_U32LE, pcm_u32le);
199. REGISTER_MUXDEMUX (PCM_U24BE, pcm_u24be);
200. REGISTER_MUXDEMUX (PCM_U24LE, pcm_u24le);
201. REGISTER_MUXDEMUX (PCM_U16BE, pcm_u16be);
202. REGISTER_MUXDEMUX (PCM_U16LE, pcm_u16le);
203. REGISTER_MUXDEMUX (PCM_U8, pcm_u8);
204. REGISTER_DEMUXER (PPM, ppm);
205. REGISTER_MUXER (PSP, psp);
206. REGISTER_DEMUXER (PVA, pva);
207. REGISTER_DEMUXER (QCP, qcp);
208. REGISTER_DEMUXER (R3D, r3d);
209. REGISTER_MUXDEMUX (RAWVIDEO, rawvideo);
210. REGISTER_DEMUXER (RL2, rl2);
211. REGISTER_MUXDEMUX (RM, rm);
212. REGISTER_MUXDEMUX (ROQ, roq);
213. REGISTER_DEMUXER (RPL, rpl);
214. REGISTER_MUXDEMUX (RSO, rso);
215. REGISTER_MUXDEMUX (RTP, rtp);
216. REGISTER_MUXDEMUX (RTSP, rtsp);
217. REGISTER_MUXDEMUX (SAP, sap);
218. REGISTER_DEMUXER (SBG, sbg);
219. REGISTER_DEMUXER (SDP, sdp);
220. #if CONFIG RTPDEC
221.     av_register_rtp_dynamic_payload_handlers();
222.     av_register_rdt_dynamic_payload_handlers();
223. #endif
224. REGISTER_DEMUXER (SEGAFILM, segafilm);
225. REGISTER_MUXER (SEGMENT, segment);
226. REGISTER_DEMUXER (SHORTEN, shorten);
227. REGISTER_DEMUXER (SIFF, siff);
228. REGISTER_DEMUXER (SMACKER, smacker);
229. REGISTER_MUXDEMUX (SMACKER, smacker);

```

```

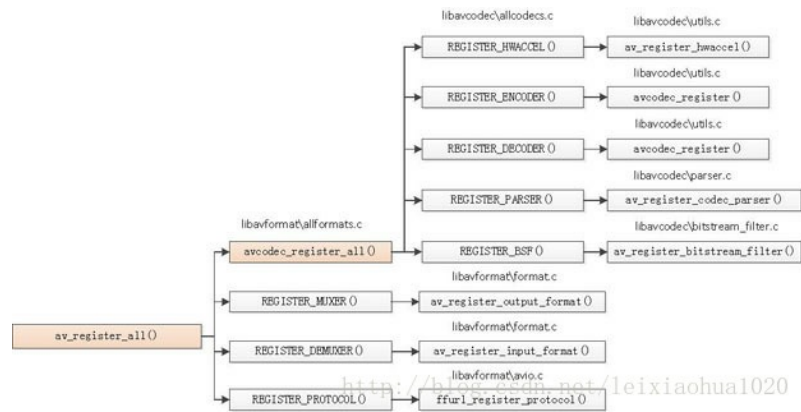
229. REGISTER_MUXDEMUX (SMJPEG, smjpeg);
230. REGISTER_DEMUXER (SOL, sol);
231. REGISTER_MUXDEMUX (SOX, sox);
232. REGISTER_MUXDEMUX (SPDIF, spdif);
233. REGISTER_MUXDEMUX (SRT, srt);
234. REGISTER_DEMUXER (STR, str);
235. REGISTER_MUXDEMUX (SWF, swf);
236. REGISTER_MUXER (TG2, tg2);
237. REGISTER_MUXER (TGP, tgp);
238. REGISTER_DEMUXER (THP, thp);
239. REGISTER_DEMUXER (TIERTEXSEQ, tiertexseq);
240. REGISTER_MUXER (MKVTIMESTAMP_V2, mkvtimestamp_v2);
241. REGISTER_DEMUXER (TMV, tmv);
242. REGISTER_MUXDEMUX (TRUEHD, truehd);
243. REGISTER_DEMUXER (TTA, tta);
244. REGISTER_DEMUXER (TXD, txd);
245. REGISTER_DEMUXER (TTY, tty);
246. REGISTER_DEMUXER (VC1, vc1);
247. REGISTER_MUXDEMUX (VC1T, vc1t);
248. REGISTER_DEMUXER (VMD, vmd);
249. REGISTER_MUXDEMUX (VOC, voc);
250. REGISTER_DEMUXER (VQF, vqf);
251. REGISTER_DEMUXER (W64, w64);
252. REGISTER_MUXDEMUX (WAV, wav);
253. REGISTER_DEMUXER (WC3, wc3);
254. REGISTER_MUXER (WEBM, webm);
255. REGISTER_DEMUXER (WSAUD, wsaud);
256. REGISTER_DEMUXER (WSVQA, wsvqa);
257. REGISTER_MUXDEMUX (WTV, wtv);
258. REGISTER_DEMUXER (WV, wv);
259. REGISTER_DEMUXER (XA, xa);
260. REGISTER_DEMUXER (XBIN, xbin);
261. REGISTER_DEMUXER (XMV, xmv);
262. REGISTER_DEMUXER (XWMA, xwma);
263. REGISTER_DEMUXER (YOP, yop);
264. REGISTER_MUXDEMUX (YUV4MPEGPIPE, yuv4mpegpipe);
265.
266. /* external libraries */
267. #if CONFIG_LIBMODPLUG
268. REGISTER_DEMUXER (LIBMODPLUG, libmodplug);
269. #endif
270. REGISTER_MUXDEMUX (LIBNUT, libnut);
271. //注册所有的Protocol (位于DEMUXER之前 (我的理解~~))
272. //文件也是一种Protocol
273. /* protocols */
274. REGISTER_PROTOCOL (APPLEHTTP, applehttp);
275. REGISTER_PROTOCOL (CACHE, cache);
276. REGISTER_PROTOCOL (CONCAT, concat);
277. REGISTER_PROTOCOL (CRYPTO, crypto);
278. REGISTER_PROTOCOL (FILE, file);
279. REGISTER_PROTOCOL (GOPHER, gopher);
280. REGISTER_PROTOCOL (HTTP, http);
281. REGISTER_PROTOCOL (HTTPPROXY, httpproxy);
282. REGISTER_PROTOCOL (HTTPS, https);
283. REGISTER_PROTOCOL (MMSH, mmsh);
284. REGISTER_PROTOCOL (MMST, mmst);
285. REGISTER_PROTOCOL (MD5, md5);
286. REGISTER_PROTOCOL (PIPE, pipe);
287. REGISTER_PROTOCOL (RTMP, rtmp);
288. //如果包含了LibRTMP
289. #if CONFIG_LIBRTMP
290. REGISTER_PROTOCOL (RTMP, rtmtpt);
291. REGISTER_PROTOCOL (RTMP, rtmpe);
292. REGISTER_PROTOCOL (RTMP, rtmpte);
293. REGISTER_PROTOCOL (RTMP, rtmps);
294. #endif
295. REGISTER_PROTOCOL (RTP, rtp);
296. REGISTER_PROTOCOL (TCP, tcp);
297. REGISTER_PROTOCOL (TLS, tls);
298. REGISTER_PROTOCOL (UDP, udp);
299. }

```

整个代码没太多可说的，首先确定是不是已经初始化过了（initialized），如果没有，就调用avcodec_register_all()注册编解码器（这个先不分析），然后就是注册，注册，注册...直到完成所有注册。

PS：曾经研究过一阵子RTMP协议，以及对应的开源工程librtmp。在这里发现有一点值得注意，ffmpeg自带了RTMP协议的支持，只有使用 rtmtpt://, rtmpe://, rtmpte://等的时候才会使用librtmp库。

函数调用关系图如下图所示。av_register_all()调用了avcodec_register_all()。avcodec_register_all()注册了和编解码器有关的组件：硬件加速器，解码器，编码器，Parser，Bitstream Filter。av_register_all()除了调用avcodec_register_all()之外，还注册了复用器，解复用器，协议处理器。



下面附上复用器，解复用器，协议处理器的代码。

注册复用器的函数是av_register_output_format()。

```
[cpp]
1. void av_register_output_format(AVOutputFormat *format)
2. {
3.     AVOutputFormat **p;
4.     p = &first_iformat;
5.     while (*p != NULL) p = &(*p)->next;
6.     *p = format;
7.     format->next = NULL;
8. }
```

注册解复用器的函数是av_register_input_format()。

```
[cpp]
1. void av_register_input_format(AVInputFormat *format)
2. {
3.     AVInputFormat **p;
4.     p = &first_iformat;
5.     while (*p != NULL) p = &(*p)->next;
6.     *p = format;
7.     format->next = NULL;
8. }
```

注册协议处理器的函数是ffurl_register_protocol()。

```
[cpp]
1. int ffurl_register_protocol(URLProtocol *protocol)
2. {
3.     URLProtocol **p;
4.     p = &first_protocol;
5.     while (*p)
6.         p = &(*p)->next;
7.     *p = protocol;
8.     protocol->next = NULL;
9.     return 0;
10. }
```

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/12677129>

文章标签： [ffmpeg](#) [源代码](#) [av_register_all](#) [分析](#) [复用](#)

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