原 ffmpeg 源代码简单分析 ： av_register_all()

2013年10月13日 12:41:11　阅读数：63534

===

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，编码器等。

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]

```cpp
#define REGISTER_DEMUXER(X,x) { \
    extern AVInputFormat ff_##x##_demuxer; \
    if(CONFIG_##X##_DEMUXER) av_register_input_format(&ff_##x##_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]

```cpp
extern AVInputFormat ff_aac_demuxer;
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]

```cpp
void av_register_input_format(AVInputFormat *format)
{
    AVInputFormat **p;
    p = &first_iformat;
    while (*p != NULL) p = &(*p)->next;
    *p = format;
    format->next = NULL;
}
```

这段代码是比较容易理解的，首先先提一点，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]

```cpp
/** head of registered input format linked list */
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]

```cpp
#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]

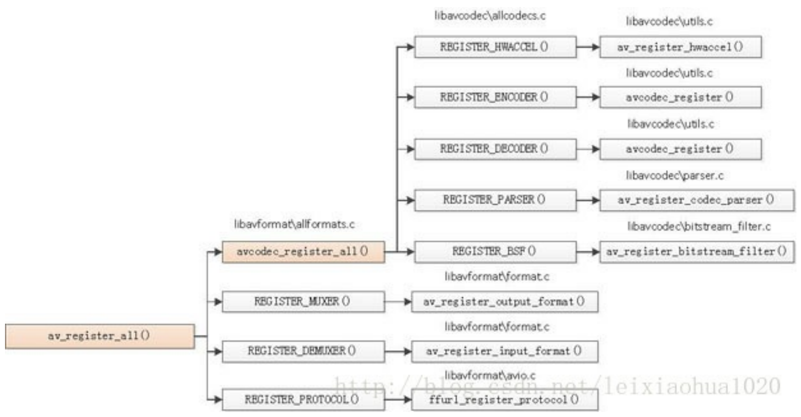
```cpp
/*
 *雷霄骅
 *leixiaohua1020@126.com
 *中国传媒大学/数字电视技术
 */
/*
 * Register all the formats and protocols
 * Copyright (c) 2000, 2001, 2002 Fabrice Bellard
 *
 * This file is part of FFmpeg.
 *
 * FFmpeg is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Lesser General Public
 * License as published by the Free Software Foundation; either
 * version 2.1 of the License, or (at your option) any later version.
 *
 * FFmpeg is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Lesser General Public License for more details.
 *
 * You should have received a copy of the GNU Lesser General Public
 * License along with FFmpeg; if not, write to the Free Software
 * Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
 */
#include "avformat.h"
#include "rtp.h"
#include "rdt.h"
#include "url.h"
//定义的宏？宏的速度会快一点？注册AVOutputFormat
//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。
//感觉有点像JAva中的EL，可以随意拼接字符串
#define REGISTER_MUXER(X,x) { \
    extern AVOutputFormat ff_##x##_muxer; \
    if(CONFIG_##X##_MUXER) av_register_output_format(&ff_##x##_muxer); }
//定义的宏？宏的速度会快一点？注册AVInputFormat
#define REGISTER_DEMUXER(X,x) { \
    extern AVInputFormat ff_##x##_demuxer; \
    if(CONFIG_##X##_DEMUXER) av_register_input_format(&ff_##x##_demuxer); }
//注册函数av_register_input_format

//定义的宏？宏的速度会快一点？两个一起注册！
#define REGISTER_MUXDEMUX(X,x)  REGISTER_MUXER(X,x); REGISTER_DEMUXER(X,x)
//定义的宏？宏的速度会快一点？注册URLProtocol
//extern URLProtocol ff_##x##_protocol;
//在librtmp中，对应的就是ff_rtmp_protocol
```

```
//这样就把librtmp整合起来了
//由此可见URLProtocol的名字是固定的
#define REGISTER_PROTOCOL(X,x) { \
    extern URLProtocol ff_##x##_protocol; \
    if(CONFIG_##X##_PROTOCOL) ffurl_register_protocol(&ff_##x##_protocol, sizeof(ff_##x##_protocol)); }
//注册函数ffurl_register_protocol
void av_register_all(void)
{
    static int initialized;

    if (initialized)
        return;
    initialized = 1;
    //注册所有的codec
    avcodec_register_all();
    //注册所有的MUXER（复用器和解复用器）
    /* (de)muxers */
    REGISTER_MUXER    (A64, a64);
    REGISTER_DEMUXER  (AAC, aac);
    REGISTER_MUXDEMUX (AC3, ac3);
    REGISTER_DEMUXER  (ACT, act);
    REGISTER_DEMUXER  (ADF, adf);
    REGISTER_MUXER    (ADTS, adts);
    REGISTER_MUXDEMUX (ADX, adx);
    REGISTER_DEMUXER  (AEA, aea);
    REGISTER_MUXDEMUX (AIFF, aiff);
    REGISTER_MUXDEMUX (AMR, amr);
    REGISTER_DEMUXER  (ANM, anm);
    REGISTER_DEMUXER  (APC, apc);
    REGISTER_DEMUXER  (APE, ape);
    REGISTER_DEMUXER  (APPLEHTTP, applehttp);
    REGISTER_MUXDEMUX (ASF, asf);
    REGISTER_MUXDEMUX (ASS, ass);
    REGISTER_MUXER    (ASF_STREAM, asf_stream);
    REGISTER_MUXDEMUX (AU, au);
    REGISTER_MUXDEMUX (AVI, avi);
    REGISTER_DEMUXER  (AVISYNTH, avisynth);
    REGISTER_MUXER    (AVM2, avm2);
    REGISTER_DEMUXER  (AVS, avs);
    REGISTER_DEMUXER  (BETHSOFTVID, bethsoftvid);
    REGISTER_DEMUXER  (BFI, bfi);
    REGISTER_DEMUXER  (BINTEXT, bintext);
    REGISTER_DEMUXER  (BINK, bink);
    REGISTER_MUXDEMUX (BIT, bit);
    REGISTER_DEMUXER  (BMV, bmv);
    REGISTER_DEMUXER  (C93, c93);
    REGISTER_MUXDEMUX (CAF, caf);
    REGISTER_MUXDEMUX (CAVSVIDEO, cavsvideo);
    REGISTER_DEMUXER  (CDG, cdg);
    REGISTER_MUXER    (CRC, crc);
    REGISTER_MUXDEMUX (DAUD, daud);
    REGISTER_DEMUXER  (DFA, dfa);
    REGISTER_MUXDEMUX (DIRAC, dirac);
    REGISTER_MUXDEMUX (DNXHD, dnxhd);
    REGISTER_DEMUXER  (DSICIN, dsicin);
    REGISTER_MUXDEMUX (DTS, dts);
    REGISTER_MUXDEMUX (DV, dv);
    REGISTER_DEMUXER  (DXA, dxa);
    REGISTER_DEMUXER  (EA, ea);
    REGISTER_DEMUXER  (EA_CDATA, ea_cdata);
    REGISTER_MUXDEMUX (EAC3, eac3);
    REGISTER_MUXDEMUX (FFM, ffm);
    REGISTER_MUXDEMUX (FFMETADATA, ffmetadata);
    REGISTER_MUXDEMUX (FILMSTRIP, filmstrip);
    REGISTER_MUXDEMUX (FLAC, flac);
    REGISTER_DEMUXER  (FLIC, flic);
    REGISTER_MUXDEMUX (FLV, flv);
    REGISTER_DEMUXER  (FOURXM, fourxm);
    REGISTER_MUXER    (FRAMECRC, framecrc);
    REGISTER_MUXER    (FRAMEMD5, framemd5);
    REGISTER_MUXDEMUX (G722, g722);
    REGISTER_MUXDEMUX (G723_1, g723_1);
    REGISTER_DEMUXER  (G729, g729);
    REGISTER_MUXER    (GIF, gif);
    REGISTER_DEMUXER  (GSM, gsm);
    REGISTER_MUXDEMUX (GXF, gxf);
    REGISTER_MUXDEMUX (H261, h261);
    REGISTER_MUXDEMUX (H263, h263);
    REGISTER_MUXDEMUX (H264, h264);
    REGISTER_DEMUXER  (ICO, ico);
    REGISTER_DEMUXER  (IDCIN, idcin);
    REGISTER_DEMUXER  (IDF, idf);
    REGISTER_DEMUXER  (IFF, iff);
    REGISTER_MUXDEMUX (IMAGE2, image2);
    REGISTER_MUXDEMUX (IMAGE2PIPE, image2pipe);
    REGISTER_DEMUXER  (INGENIENT, ingenient);
    REGISTER_DEMUXER  (IPMOVIE, ipmovie);
    REGISTER_MUXER    (IPOD, ipod);
    REGISTER_MUXER    (ISMV, ismv);
    REGISTER_DEMUXER  (ISS, iss);
    REGISTER_DEMUXER  (IV8, iv8);
    REGISTER_MUXDEMUX (IVF, ivf);
    REGISTER_DEMUXER  (JV, jv);
    REGISTER_MUXDEMUX (LATM, latm);
    REGISTER_DEMUXER  (LMLM4, lmlm4);
    REGISTER_DEMUXER  (LOAS, loas);
    REGISTER_DEMUXER  (LXF, lxf);
    REGISTER_MUXDEMUX (M4V, m4v);
    REGISTER_MUXER    (MD5, md5);
    REGISTER_MUXDEMUX (MATROSKA, matroska);
    REGISTER_MUXER    (MATROSKA_AUDIO, matroska_audio);
    REGISTER_MUXDEMUX (MICRODVD, microdvd);
    REGISTER_MUXDEMUX (MJPEG, mjpeg);
    REGISTER_MUXDEMUX (MLP, mlp);
    REGISTER_DEMUXER  (MM, mm);
    REGISTER_MUXDEMUX (MMF, mmf);
    REGISTER_MUXDEMUX (MOV, mov);
    REGISTER_MUXER    (MP2, mp2);
    REGISTER_MUXDEMUX (MP3, mp3);
    REGISTER_MUXER    (MP4, mp4);
    REGISTER_DEMUXER  (MPC, mpc);
    REGISTER_DEMUXER  (MPC8, mpc8);
    REGISTER_MUXER    (MPEG1SYSTEM, mpeg1system);
    REGISTER_MUXER    (MPEG1VCD, mpeg1vcd);
    REGISTER_MUXER    (MPEG1VIDEO, mpeg1video);
    REGISTER_MUXER    (MPEG2DVD, mpeg2dvd);
    REGISTER_MUXER    (MPEG2SVCD, mpeg2svcd);
    REGISTER_MUXER    (MPEG2VIDEO, mpeg2video);
    REGISTER_MUXER    (MPEG2VOB, mpeg2vob);
    REGISTER_DEMUXER  (MPEGPS, mpegps);
    REGISTER_MUXDEMUX (MPEGTS, mpegts);
    REGISTER_DEMUXER  (MPEGTSRAW, mpegtsraw);
    REGISTER_DEMUXER  (MPEGVIDEO, mpegvideo);
    REGISTER_MUXER    (MPJPEG, mpjpeg);
    REGISTER_DEMUXER  (MSNWC_TCP, msnwc_tcp);
    REGISTER_DEMUXER  (MTV, mtv);
    REGISTER_DEMUXER  (MVI, mvi);
    REGISTER_MUXDEMUX (MXF, mxf);
    REGISTER_MUXER    (MXF_D10, mxf_d10);
    REGISTER_DEMUXER  (MXG, mxg);
    REGISTER_DEMUXER  (NC, nc);
    REGISTER_DEMUXER  (NSV, nsv);
    REGISTER_MUXER    (NULL, null);
    REGISTER_MUXDEMUX (NUT, nut);
    REGISTER_DEMUXER  (NUV, nuv);
    REGISTER_MUXDEMUX (OGG, ogg);
    REGISTER_MUXDEMUX (OMA, oma);
    REGISTER_MUXDEMUX (PCM_ALAW,  pcm_alaw);
    REGISTER_MUXDEMUX (PCM_MULAW, pcm_mulaw);
    REGISTER_MUXDEMUX (PCM_F64BE, pcm_f64be);
    REGISTER_MUXDEMUX (PCM_F64LE, pcm_f64le);
    REGISTER_MUXDEMUX (PCM_F32BE, pcm_f32be);
    REGISTER_MUXDEMUX (PCM_F32LE, pcm_f32le);
    REGISTER_MUXDEMUX (PCM_S32BE, pcm_s32be);
    REGISTER_MUXDEMUX (PCM_S32LE, pcm_s32le);
    REGISTER_MUXDEMUX (PCM_S24BE, pcm_s24be);
    REGISTER_MUXDEMUX (PCM_S24LE, pcm_s24le);
    REGISTER_MUXDEMUX (PCM_S16BE, pcm_s16be);
    REGISTER_MUXDEMUX (PCM_S16LE, pcm_s16le);
    REGISTER_MUXDEMUX (PCM_S8,    pcm_s8);
    REGISTER_MUXDEMUX (PCM_U32BE, pcm_u32be);
    REGISTER_MUXDEMUX (PCM_U32LE, pcm_u32le);
    REGISTER_MUXDEMUX (PCM_U24BE, pcm_u24be);
    REGISTER_MUXDEMUX (PCM_U24LE, pcm_u24le);
    REGISTER_MUXDEMUX (PCM_U16BE, pcm_u16be);
    REGISTER_MUXDEMUX (PCM_U16LE, pcm_u16le);
    REGISTER_MUXDEMUX (PCM_U8,    pcm_u8);
    REGISTER_DEMUXER  (PMP, pmp);
    REGISTER_MUXER    (PSP, psp);
    REGISTER_DEMUXER  (PVA, pva);
    REGISTER_DEMUXER  (QCP, qcp);
    REGISTER_DEMUXER  (R3D, r3d);
    REGISTER_MUXDEMUX (RAWVIDEO, rawvideo);
    REGISTER_DEMUXER  (RL2, rl2);
    REGISTER_MUXDEMUX (RM, rm);
    REGISTER_MUXDEMUX (ROQ, roq);
    REGISTER_DEMUXER  (RPL, rpl);
    REGISTER_MUXDEMUX (RSO, rso);
    REGISTER_MUXDEMUX (RTP, rtp);
    REGISTER_MUXDEMUX (RTSP, rtsp);
    REGISTER_MUXDEMUX (SAP, sap);
    REGISTER_DEMUXER  (SBG, sbg);
    REGISTER_DEMUXER  (SDP, sdp);
#if CONFIG_RTPDEC
    av_register_rtp_dynamic_payload_handlers();
    av_register_rdt_dynamic_payload_handlers();
#endif
    REGISTER_DEMUXER  (SEGAFILM, segafilm);
    REGISTER_MUXER    (SEGMENT, segment);
    REGISTER_DEMUXER  (SHORTEN, shorten);
    REGISTER_DEMUXER  (SIFF, siff);
    REGISTER_DEMUXER  (SMACKER, smacker);
    REGISTER_MUXDEMUX (SMJPEG, smjpeg);
    REGISTER_DEMUXER  (SOL, sol);
    REGISTER_MUXDEMUX (SOX, sox);
    REGISTER_MUXDEMUX (SPDIF, spdif);
    REGISTER_MUXDEMUX (SRT, srt);
    REGISTER_DEMUXER  (STR, str);
    REGISTER_MUXDEMUX (SWF, swf);
    REGISTER_MUXER    (TG2, tg2);
    REGISTER_MUXER    (TGP, tgp);
    REGISTER_DEMUXER  (THP, thp);
    REGISTER_DEMUXER  (TIERTEXSEQ, tiertexseq);
    REGISTER_MUXER    (MKVTIMESTAMP_V2, mkvtimestamp_v2);
    REGISTER_DEMUXER  (TMV, tmv);
    REGISTER_MUXDEMUX (TRUEHD, truehd);
    REGISTER_DEMUXER  (TTA, tta);
    REGISTER_DEMUXER  (TXD, txd);
    REGISTER_DEMUXER  (TTY, tty);
    REGISTER_DEMUXER  (VC1, vc1);
    REGISTER_MUXDEMUX (VC1T, vc1t);
    REGISTER_DEMUXER  (VMD, vmd);
    REGISTER_MUXDEMUX (VOC, voc);
    REGISTER_DEMUXER  (VQF, vqf);
    REGISTER_DEMUXER  (W64, w64);
    REGISTER_MUXDEMUX (WAV, wav);
    REGISTER_DEMUXER  (WC3, wc3);
    REGISTER_MUXER    (WEBM, webm);
    REGISTER_DEMUXER  (WSAUD, wsaud);
    REGISTER_DEMUXER  (WSVQA, wsvqa);
    REGISTER_MUXDEMUX (WTV, wtv);
    REGISTER_DEMUXER  (WV, wv);
    REGISTER_DEMUXER  (XA, xa);
    REGISTER_DEMUXER  (XBIN, xbin);
    REGISTER_DEMUXER  (XMV, xmv);
    REGISTER_DEMUXER  (XWMA, xwma);
    REGISTER_DEMUXER  (YOP, yop);
    REGISTER_MUXDEMUX (YUV4MPEGPIPE, yuv4mpegpipe);

    /* external libraries */
#if CONFIG_LIBMODPLUG
    REGISTER_DEMUXER  (LIBMODPLUG, libmodplug);
#endif
    REGISTER_MUXDEMUX (LIBNUT, libnut);
    //注册所有的Protocol（位于DEMUXER之前（我的理解~~））
    //文件也是一种Protocol
    /* protocols */
    REGISTER_PROTOCOL (APPLEHTTP, applehttp);
    REGISTER_PROTOCOL (CACHE, cache);
    REGISTER_PROTOCOL (CONCAT, concat);
    REGISTER_PROTOCOL (CRYPTO, crypto);
    REGISTER_PROTOCOL (FILE, file);
    REGISTER_PROTOCOL (GOPHER, gopher);
    REGISTER_PROTOCOL (HTTP, http);
    REGISTER_PROTOCOL (HTTPPROXY, httpproxy);
    REGISTER_PROTOCOL (HTTPS, https);
    REGISTER_PROTOCOL (MMSH, mmsh);
    REGISTER_PROTOCOL (MMST, mmst);
    REGISTER_PROTOCOL (MD5,  md5);
    REGISTER_PROTOCOL (PIPE, pipe);
    REGISTER_PROTOCOL (RTMP, rtmp);
//如果包含了LibRTMP
#if CONFIG_LIBRTMP
    REGISTER_PROTOCOL (RTMP, rtmpt);
    REGISTER_PROTOCOL (RTMP, rtmpe);
    REGISTER_PROTOCOL (RTMP, rtmpte);
    REGISTER_PROTOCOL (RTMP, rtmps);
#endif
    REGISTER_PROTOCOL (RTP, rtp);
    REGISTER_PROTOCOL (TCP, tcp);
    REGISTER_PROTOCOL (TLS, tls);
    REGISTER_PROTOCOL (UDP, udp);
}
```

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 rtmpt://, 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]

```cpp
void av_register_output_format(AVOutputFormat *format)
{
    AVOutputFormat **p;
    p = &first_oformat;
    while (*p != NULL) p = &(*p)->next;
    *p = format;
    format->next = NULL;
}
```

注册解复用器的函数是av_register_input_format()。

[cpp]

```cpp
void av_register_input_format(AVInputFormat *format)
{
    AVInputFormat **p;
    p = &first_iformat;
    while (*p != NULL) p = &(*p)->next;
    *p = format;
    format->next = NULL;
}
```

注册协议处理器的函数是ffurl_register_protocol()。

[cpp]

```cpp
int ffurl_register_protocol(URLProtocol *protocol)
{
    URLProtocol **p;
    p = &first_protocol;
    while (*p)
        p = &(*p)->next;
    *p             = protocol;
    protocol->next = NULL;
    return 0;
}
```

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