全志科技
Allwinner Technology

音视频解码库项目

流媒体协议库 API 说明

文档履历

版本号	日期	制/修订人	内容描述
V0.1	2014-1-24		初稿
V0.2	2014-9-12		修订 1、完善 2.2 相关术语描述 2、删除原 2.3.模块配置介绍 3、增加 2.3 源码结构介绍 4、增加模块体系结构设计内容 5、修改 4.1 接口函数表格，增加内容描述 6、4.1.8 增加 CdxStreamWrite，CdxStreamGetMetaData 接口函数介绍，删除 CdxStreamForceStop 描述（在 control 中介绍） 7、增加 5.1 数据结构 8、修改 5.1.1.struct CdxStreamOpsS 描述,增加 write、getMetaData 9、修改 struct CdxDataSourceS 10、增加 enum CdxIOStateE、enum CdxStreamCommandE、struct StreamCacheStateS、struct CdxStreamNodeS 数据结构。 11、修改 6.实现样例
V0.3	2015-05-28		去掉原先 CdxStreamOpen，增加 4.1.1,4.1.2; 修改 6.实现样例
V1.0	2015-05-28		Release 版
V1.1	2015-10-12		增加 7.配置说明
V1.2	2016-11-03		添加 CdxStreamOpen 接口说明 添加 5.1.9、5.1.10

目 录

1. 概述

1.1. 编写目的

设计媒体播放器 CedarX 的对所有协议类型的 Stream（流媒体，下同）访问的统一接口。指导具体协议类型 Stream 的开发、使用和后续维护。

1.2. 适用范围

A80/A83/H3/H8 等各个芯片平台的 Android 系统 SDK 和 Linux SDK。

1.3. 相关人员

开发和维护 Stream 协议的相关人员，二次开发人员，相关依赖模块（如 Parser--媒体文件格式解析库）开发和维护人员。

2. 模块介绍

2.1. 功能介绍

各 stream 协议实现本文档所描述的接口，并且遵循接口特性描述，实现对应 stream 协议数据里的访问。本项目支持的 stream 协议类型包括：本地文件、文件描述符、RTSP、UDP、RTP、HTTP、SSL、TCP、RTMP、MMS、MMSH、MMST、MMSHTTP、AES、BDMV 等。各个 stream 协议按照提供的注册方式进行注册，以后新增的协议也如此，以便扩展。

2.2. 相关术语介绍

stream：数据流，包括本地文件和流媒体文件。它是播放的对象，同时也是 Parser 解封装的对象。

Probe：探测，从 stream 中读取一段数据以探测其封装格式。

2.3. 源码结构介绍

```
STREAM/
├── Android.mk
├── config.mk   //全局的配置文件，子目录以下的 Android.mk 都需要包含该文件
├── include
│   └── CdxStream.h   //本模块对外的头文件
├── base
│   └── CdxStream.c   //实例化一个 Stream
├── aes
│   └── ……  //aes stream 实现代码
├── file
│   └── ……  //file stream 实现代码
├── mms
│   └── ……  //mms stream 实现代码
├── rtsp
│   └── ……  //rtsp stream 实现代码
├── http
│   └── ……  //http stream 实现代码
├── tcp
│   └── ……  //tcp stream 实现代码
│
│…… //其它 stream
```

3. 模块体系结构设计

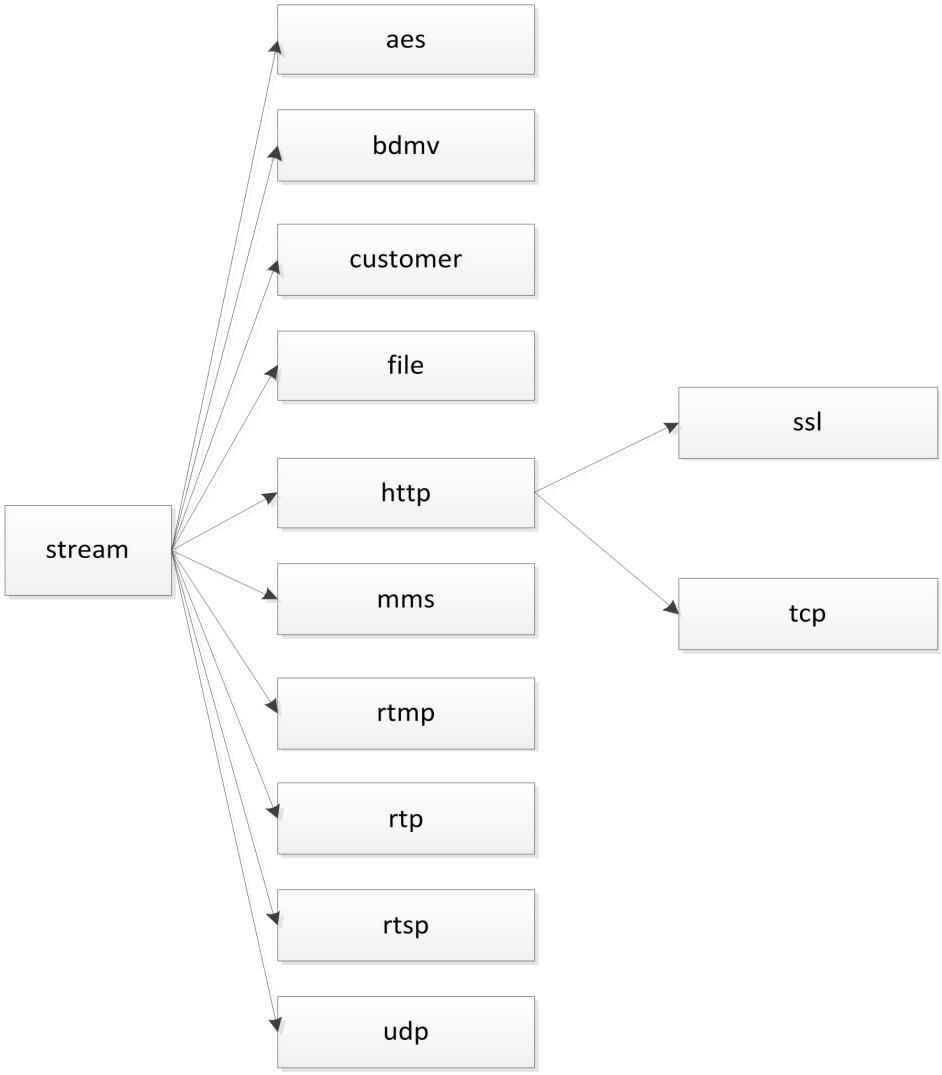


图 1 流媒体协议库模块体系结构图

图 1 所示为流媒体协议库模块的大致体系结构，上层（parser 库）通过 CdxDataSourceT 成员的 uri 所带的 scheme，选择打开相应 stream。理论上来说，上层可以打开任意 stream，stream 也可以打开任意 stream，只要根据需要构造带上相应 scheme 的 uri。例如目前实现的 aes stream 可以通过 CdxStreamOpen 打开一个 http stream，读取所需数据进行相应操作；当然也可以打开一个 file stream 进行相应数据访问。

根据不同 scheme 选择打开 stream 的设计使得上层（parser 库）选择调用 stream、stream 调用 stream 更加灵活，也在一定程度上实现代码复用，减少重复代码。

4. 接口和流程设计

4.1. 接口函数

流媒体模块接口函数	
CdxStreamOpen	调用 CdxStreamCreate 和 CdxStreamConnect 创建和连接 stream
CdxStreamCreate	创建一个 Stream 对象。
CdxStreamConnect	连接一个 stream。
CdxStreamGetProbeData	获取供 probe 的数据。
CdxStreamRead	从 stream 对象读取数据。
CdxStreamClose	关闭 stream 对象，释放资源。
CdxStreamGetIoState	获取 stream 的 IO 状态（CDX_IO_STATE_OK、CDX_IO_STATE_ERROR 等）。
CdxStreamAttribute	获取 stream 的属性信息，包括是否网络流，是否可以 seek。
CdxStreamControl	命令控制函数，可以扩展。
CdxStreamWrite	向 stream 对象写数据。
CdxStreamGetMetaData	获取 stream 的元数据，如 uri 等，可以扩展。
CdxStreamSeek	根据 whence seek 到指定位置。
CdxStreamSeekToTime	Seek 到指定时间点。
CdxStreamEos	是否读到尾部。
CdxStreamTell	获取当前的读写位置。
CdxStreamSize	获取 stream 的 size。

需要注意的是，以上表格中从 CdxStreamCreate 到 CdxStreamControl 函数是各个 stream 必须实现的，CdxStreamWrite 以下的根据需要实现。当要添加新的 stream 时，需要在 gStreamList 结构体数组中按照格式填写相应 stream 的 CdxStreamCreatorT 以完成注册。

当需要对 stream 进行读写操作时，首先需要打开一个 stream。通过匹配具体 stream 名和 gStreamList 数组中的 stream 名，选择所需的 stream 打开。打开成功后，便可以进行 CdxStreamGetProbeData 确定具体封装类型、CdxStreamRead 读取指定长度数据、CdxStreamSeek 跳到指定位置等操作。下面对各个接口分别进行介绍。

4.1.1. CdxStreamOpen

函数原型	int CdxStreamOpen(CdxDataSourceT *source, pthread_mutex_t *mutex, cdx_bool *exit,CdxStreamT **stream, ContorlTask *streamTasks);
功能	调用 CdxStreamCreate 创建一个 stream，并且下发了 streamTasks 控制命令之后再去连接 stream
参数	source ： 包含流媒体对应 URI 信息和其他头部信息 mutex ： 线程互斥锁 exit： 退出控制标志 stream： 创建成功返回的 stream 句柄

	streamTasks：连接前的控制命令
返回值	成功：0 失败：-1
调用说明	是阻塞函数，connect 成功后才能进行其他操作。

4.1.2. CdxStreamCreate

函数原型	CdxStreamT *(*create)(CdxDataSourceT * source)
功能	创建一个 stream 对象
参数	source ： 包含流媒体对应 URI 信息和其他头部信息
返回值	成功：返回 stream 句柄 失败：返回 NULL
调用说明	NA

4.1.3. CdxStreamConnect

函数原型	cdx_int32 CdxStreamConnect(CdxStreamT *stream)
功能	连接 stream
参数	stream ： 要连接的 stream
返回值	成功：0 失败：-1
调用说明	是阻塞函数，connect 成功后才能进行其他操作。

4.1.4. CdxStreamGetProbeData

函数原型	CdxStreamProbeDataT *CdxStreamGetProbeData(CdxStreamT *stream)
功能	获取该 stream 供 probe 的数据，以确定封装格式。
参数	stream ： 句柄
返回值	成功：供 probe 的数据，包括长度和数据，调用者不应该对该数据做写的操作 失败：返回 NULL
调用说明	Open stream 成功后才能得到 probe 数据

4.1.5. CdxStreamRead

函数原型	cdx_int32 CdxStreamRead(CdxStreamT *stream, void *buf, cdx_int32 len)
功能	从 stream 读取数据
参数	Stream：句柄

	Buf：存放数据内存块首地址 Len：需要读取数据的长度
返回值	成功：读取到的数据长度 失败：返回-1，通过 getIoState 获取错误原因
调用说明	1、成功时会往 buf 内存填写所读到的数据 2、是阻塞函数，可以调用 forceStop 终止执行，forceStop 功能由下面的宏实现：#define CdxStreamForceStop(stream) \ (CdxStreamControl(stream, STREAM_CMD_SET_FORCESTOP, NULL))，具体参考 CdxStreamControl 的实现。

4.1.6. CdxStreamClose

函数原型	cdx_int32 CdxStreamClose(CdxStreamT *stream)
功能	关闭 stream 对象
参数	Stream：句柄
返回值	成功：返回 0 失败：返回-1，close 失败是比较致命的错误了，得人工排查系统
调用说明	NA

4.1.7. CdxStreamGetIoState

函数原型	cdx_int32 CdxStreamGetIoState(CdxStreamT *stream)
功能	获取对 Stream IO 操作的错误原因
参数	Stream：句柄
返回值	错误码，详细见 CdxIOStateE
调用说明	每次 IO 操作都会重置错误码,应该在每一次错误发生之后判断一下错误类型

4.1.8. CdxStreamAttribute

函数原型	cdx_uint32 CdxStreamAttribute(CdxStreamT *stream)
功能	获取 stream 的属性信息,包括:是否网络流,是否可 seek,是否可 seekToTime,其他未使用标志位可通过该接口做扩展使用
参数	Stream：句柄
返回值	Int 类型，是各标志位相或的集合
调用说明	返回值和以下变量做相与运算获取指定属性信息 #define CDX_STREAM_FLAG_SEEK 0x01U /* seek to pos*/ #define CDX_STREAM_FLAG_STT 0x02U /*seek to time*/ #define CDX_STREAM_FLAG_NET 0x04U /*net work stream*/

4.1.9. CdxStreamControl

函数原型	cdx_int32 CdxStreamControl(CdxStreamT *stream, cdx_int32 cmd, void *param)
功能	向 stream 发控制命令

参数	Stream：句柄 Cmd：发送命令，所有命令类型见 CdxStreamCommandE Param：参数，不同命令，参数不同
返回值	成功：返回 0 失败：返回-1，通过 getIoState 获取错误原因
调用说明	不同 stream 协议可做特殊的扩展

4.1.10. CdxStreamWrite

函数原型	cdx_int32 CdxStreamWrite(CdxStreamT *stream, void *buf, cdx_int32 len)
功能	向 stream 写入数据
参数	Stream：句柄 buf：需要写入的数据内存块首地址 len：要写入数据长度
返回值	成功：返回 0 失败：返回-1，通过 getIoState 获取错误原因
调用说明	是阻塞函数，可以调用 forceStop 终止执行，forceStop 参见 CdxStreamRead 的调用说明。

4.1.11. CdxStreamGetMetaData

函数原型	cdx_int32 CdxStreamGetMetaData(CdxStreamT *stream, const cdx_char *key, cdx_void **pVal)
功能	获取 stream 的元数据，如 uri 等。
参数	Stream：句柄 key：要获取的元数据标识，如“uri”、“extra-data”等，可以扩展 pVal：获取到的数据存入的目标地址
返回值	成功：返回 0 失败：返回-1，通过 getIoState 获取错误原因
调用说明	不同 stream 协议可做特殊的扩展，使用时按需对 pVal 做类型强制转换。

4.1.12. CdxStreamSeek

函数原型	cdx_int32 CdxStreamSeek(CdxStreamT *stream, cdx_int64 offset, cdx_int32 whence)
功能	改变对 stream 的读写位置（以偏移做标识）
参数	Stream：句柄 Offset：偏移的大小 Whence：STREAM_SEEK_SET，从距文件开头 offset 位移量为新的读写位置；STREAM_SEEK_CUR，以目前的读写位置往后增加 offset 个位移量；STREAM_SEEK_END，将读写位置指向文件尾后再加上 offset 个偏移量。
返回值	成功：返回 0 失败：返回-1，通过 getIOState 获取错误原因
调用说明	1、非强制实现接口，调用前，需要查询 stream 属性，判断是否支持 seek。

	2、是阻塞函数，可以调用 forceStop 终止执行，forceStop 参见 CdxStreamRead 的调用说明。

4.1.13. CdxStreamSeekToTime

函数原型	cdx_int32 CdxStreamSeekToTime(CdxStreamT *stream, cdx_int64 timeUs)
功能	改变对 stream 的读写位置（以时间做标识）
参数	Stream：句柄 timeUs：时间点（以微秒做单位）
返回值	成功：返回 0 失败：返回-1，通过 getIoState 获取错误原因
调用说明	非强制实现接口，调用前，需要查询 stream 属性，判断是否支持 seekToTime

4.1.14. CdxStreamEos

函数原型	cdx_bool CdxStreamEos(CdxStreamT *stream)
功能	判断是否读到尾部
参数	Stream：句柄
返回值	读到尾部返回 CDX_TRUE 否则返回 CDX_FALSE
调用说明	非强制实现接口，对于没有实现的 stream 返回 CDX_FALSE

4.1.15. CdxStreamTell

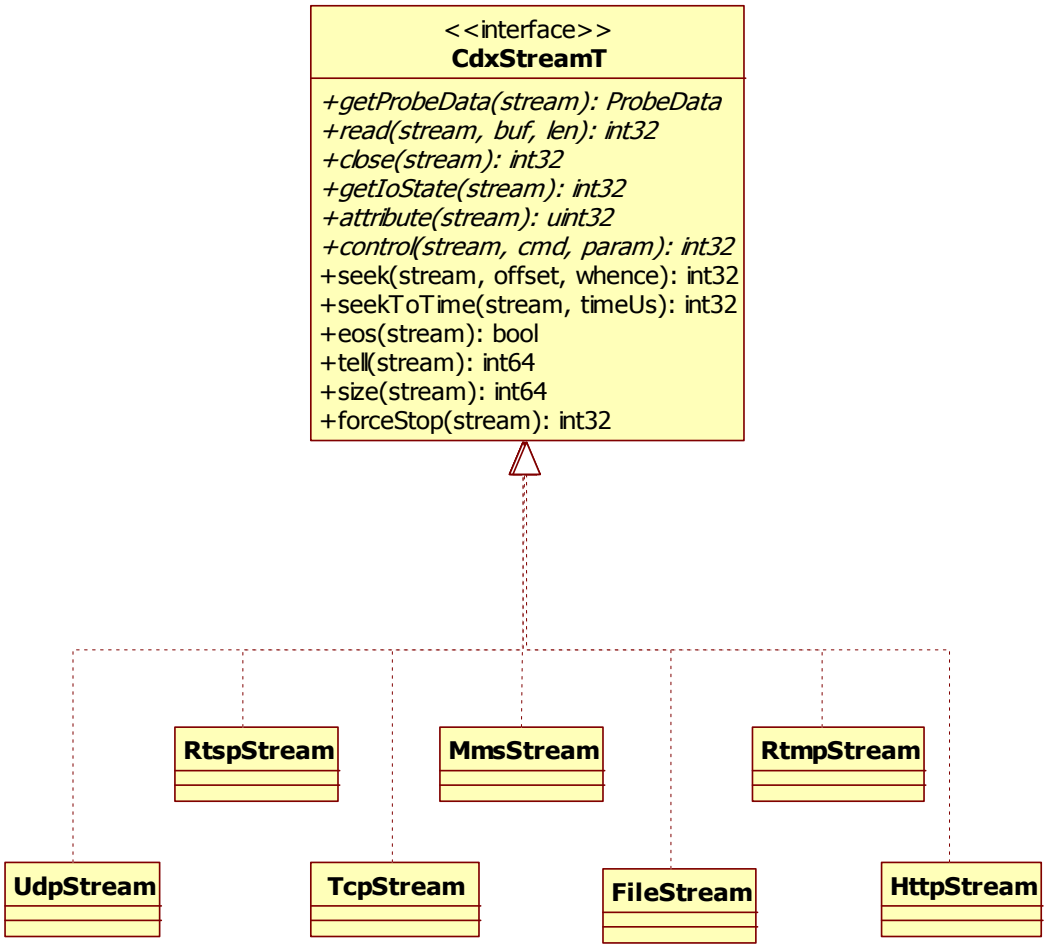
函数原型	cdx_int64 CdxStreamTell(CdxStreamT *stream)
功能	获取 stream 当前的读写位置
参数	Stream：句柄
返回值	成功：返回当前的读写位置（以距离文件开始的偏移标识） 失败：返回-1
调用说明	非强制实现接口

4.1.16. CdxStreamSize

函数原型	cdx_int64 CdxStreamSize(CdxStreamT *stream)
功能	获取 stream 的总大小
参数	Stream：句柄
返回值	成功：返回该 stream 的总大小 失败：返回-1
调用说明	1、非强制实现接口，没有实现的 stream 返回-1 2、对于没有 stream 总大小的流，如直播流，返回-1

4.2. 类关系图

CedarX 默认支持，HTTP、RTSP、RTMP、MMS、UDP、TCP 和本地文件等 stream，类关系图如下：



5. 数据结构设计

5.1. 数据结构

数据结构	
struct CdxStreamOpsS	Stream 的接口定义。
struct CdxStreamProbeDataS	描述 stream 供 probe 的数据。
struct CdxDataSourceS	描述 stream 的资源访问信息。
struct CdxStreamCreatorS	各 stream 的创建接口。
enum CdxIOStateE	描述 stream 的 IO 状态。
enum CdxStreamCommandE	Stream 用到的控制命令。
struct StreamCacheStateS	描述 stream 的缓冲状态。
struct CdxStreamNodeS	各 stream 的注册节点。
struct CallBack	stream 回调函数结构体
struct ContorlTaskS	Stream 控制命令链表

5.1.1. struct CdxStreamOpsS

名称	struct CdxStreamOpsS	
功能描述	Stream 的接口定义	
方法		
getProbeData	原型	CdxStreamProbeDataT *(*getProbeData)(CdxStreamT * /*stream*/);
	参数	Stream：句柄
	返回值	成功：返回 CdxStreamProbeDataT 的指针 失败：返回 NULL
	功能	获取 stream 供 probe 使用的数据
read	原型	cdx_int32 (*read)(CdxStreamT * /*stream*/, void * /*buf*/, cdx_uint32 /*len*/);
	参数	Stream：句柄 Buf：读取到的数据存放的地址 Len：需要读取的数据大小
	返回值	成功：返回读到的数据大小 失败：返回-1
	功能	从 stream 读取数据
close	原型	cdx_int32 (*close)(CdxStreamT * /*stream*/);
	参数	Stream：句柄
	返回值	成功：返回 0 失败：返回-1
	功能	关闭 stream
getIoState	原型	cdx_int32 (*getIOState)(CdxStreamT * /*stream*/);
	参数	Stream：句柄

	返回值	返回错误码
	功能	获取上一次 IO 操作的错误码
attribute	原型	cdx_uint32 (*attribute)(CdxStreamT * /*stream*/);
	参数	Stream：句柄
	返回值	返回该 stream 的属性标志
	功能	获取 stream 的属性标志，判断是否网络流，是否支持 seek 等
control	原型	cdx_int32 (*control)(CdxStreamT * /*stream*/, cdx_int32 /*cmd*/, void * /*param*/);
	参数	Stream：句柄 Cmd：命令 Param：参数
	返回值	成功：返回 0 失败：返回-1
	功能	向 stream 发控制命令
write	原型	cdx_int32 (*write)(CdxStreamT */*stream*/, void * /*buf*/, cdx_uint32 /*len*/);
	参数	Stream：句柄 Buf：要写入的数据存放地址 Len：要写入的数据长度
	返回值	成功：返回 0 失败：返回-1
	功能	向 stream 写入数据
getMetaData	原型	cdx_int32 (*getMetaData)(CdxStreamT */*stream*/, const cdx_char * /*key*/, cdx_void ** /*pVal*/);
	参数	Stream：句柄 Key：要获取的元数据标识 pVal：存放获取元数据的地址
	返回值	成功：返回 0 失败：返回-1
	功能	获取 stream 的元数据
seek	原型	cdx_int32 (*seek)(CdxStreamT * /*stream*/, cdx_int64 /*offset*/, cdx_int32 /*whence*/);
	参数	Stream：句柄 Offset：偏移大小 Whence：偏移起始位置
	返回值	成功：返回 0 失败：返回-1
	功能	改变 stream 的读写位置
seekToTime	原型	cdx_int32 (*seekToTime)(CdxStreamT */*stream*/, cdx_int64 /*time us*/);
	参数	Stream：句柄 timeUs：时间

	返回值	成功：返回 0 失败：返回-1
	功能	Seek 到指定时间点
eos	原型	cdx_bool (*eos)(CdxStreamT * /*stream*/);
	参数	Stream：句柄
	返回值	已经读到尾部则返回 CDX_TRUE 其他情况返回 CDX_FALSE
	功能	判断是否读到 stream 的尾部
tell	原型	cdx_int64 (*tell)(CdxStreamT * /*stream*/)
	参数	Stream：句柄
	返回值	成功：返回当前读写的位置 失败：返回-1
	功能	获取 stream 的当前读写位置
size	原型	cdx_int64 (*size)(CdxStreamT * /*stream*/);
	参数	Stream：句柄
	返回值	成功：返回 stream 大小 失败：返回-1
	功能	获取 stream 大小

5.1.2. struct CdxStreamProbeDataS

名称	struct CdxStreamProbeDataS		
功能描述	描述 stream 供 probe 的数据		
属性	类型	初始值	描述
buf	cdx_char *	NULL	供 probe 的数据地址
len	cdx_uint32	0	供 probe 的数据大小，目前最大为 128KBytes
uri	cdx_char*	NULL	stream 的 url 字符串

5.1.3. struct CdxDataSourceS

名称	struct CdxDataSourceS		
功能描述	描述 stream 的资源访问信息		
属性	类型	初始值	描述
uri	cdx_char *	N/A	资源信息的标识符，一般是 URL
extraData	cdx_void *	N/A	供不同的 stream 传输不同类型的数据
extraDataType	enum DSExtraDataTypeE	N/A	表示 extraData 的类型，目前有以下几种，可扩展： EXTRA_DATA_UNKNOWN：extraData 未知 EXTRA_DATA_HTTP_HEADER： extraData 是 http 头域 EXTRA_DATA_RTP：extraData 用于 rtp

			EXTRA_DATA_AES：extraData 用于 aes EXTRA_DATA_BDMV：extraData 用于 BDMV

5.1.4. struct CdxStreamCreatorS

名称	struct CdxStreamCreatorS	
功能描述	各 stream 的创建接口	
方法		
open	原型	CdxStreamT *(*open)(CdxDataSourceT * /*dataSource*/)
	参数	dataSource：stream 的资源访问信息
	返回值	成功：返回 stream 句柄 失败：返回 NULL
	功能	打开一个 stream，为了保证非阻塞特性，要求实现函数不能有延时操作

5.1.5. enum CdxIOStateE

名称	enum CdxIOStateE
功能	描述 stream 的 IO 状态
取值	描述
CDX_IO_STATE_OK	表示 stream 的数据访问状态正常。
CDX_IO_STATE_INVALID	表示 stream 还处于初始化阶段，尚无法读取数据。
CDX_IO_STATE_ERROR	表示 stream IO 状态出错，无法正常访问数据。
CDX_IO_STATE_EOS	表示 stream 读到文件末。
CDX_IO_STATE_CLEAR	表示要重新定位 stream 位置，用于 mms。

5.1.6. enum CdxStreamCommandE

名称	enum CdxStreamCommandE
功能	描述 stream 的控制命令，作为 control 的 cmd 参数使用。
取值	描述
STREAM_CMD_GET_DURATION	用于获取文件时长。此时 control 的 param 参数为 int64_t * 类型，获取的时长为*（int64_t*）param。
STREAM_CMD_READ_NOBLOCK	非阻塞读取数据，读到数据便立即返回以区别 read 接口阻塞读。目前只有 ssl、tcp stream 实现该命令。此时 control 函数的 param 参数置 NULL。
STREAM_CMD_GET_SOCKRECVBUFLEN	获取套接字接收缓冲区大小的命令。在 tcp stream 中实现。此时 control 的 param 参数为 cdx_int32 类型，获取的大小为*（cdx_int32*）param。

STREAM_CMD_GET_CACHESTATE	获取 stream 的缓冲状态，此时 control 函数的 param 参数为 struct StreamCacheStateS* 类型，详见 StreamCacheStateS。
STREAM_CMD_SET_FORCESTOP	该命令实现从 stream 的阻塞操作中强制退出，一般用于跳播、退出播放时快速从 read、seek 等阻塞操作中快速退出，以便优化操作体验。该命令和 STREAM_CMD_CLR_FORCESTOP 作用相反，需要通过 STREAM_CMD_CLR_FORCESTOP 清除标志。
STREAM_CMD_CLR_FORCESTOP	清除 forcestop 标志。
STREAM_CMD_RESET_STREAM	用于 mms 重新定位 stream 位置。
STREAM_CMD_EXT_IO_OPERATION	该命令实现 bdmv 的访问控制。

5.1.7. struct StreamCacheStateS

名称	struct StreamCacheStateS		
功能描述	描述 stream 的缓冲区状态。		
属性	类型	初始值	描述
nCacheCapacity	cdx_int32	N/A	表示 stream 的缓冲区大小。
nCacheSize	cdx_int32	N/A	表示已缓冲数据大小。
nBandwidthKbps	cdx_int32	N/A	表示下载数据的带宽大小，单位是 bps。
nPercentage	cdx_int32	N/A	表示下载到文件位置的百分比。

5.1.8. struct CdxStreamNodeS

名称	struct CdxStreamNodeS		
功能描述	描述 stream 的注册节点，新加的 stream 需要填入该结构体以完成注册。		
属性	类型	初始值	描述
scheme	cdx_int32	N/A	表示 stream 的标识，如 file、mms 等。
creator	cdx_int32	N/A	表示 stream 的创建接口。

5.1.9. struct CallBack

名称	struct CallBack		
功能描述	描述 stream 和 parser 模块的回调函数结构体。		
属性	类型	初始值	描述
callback	ParserCallback	N/A	回调函数句柄。
pUserData	void*	N/A	回调实体的句柄。

5.1.10. struct ContorlTaskS

名称	struct ContorlTaskS		
功能描述	stream 和 parser 的控制命令链表		
属性	类型	初始值	描述
cmd	cdx_int32	N/A	控制命令 id
param	void*	N/A	该命令的参数
next	ContorlTask *	N/A	链表下一个命令的节点指针

6. 实现样例（本地文件流）

1）实现接口

```
static CdxStreamProbeDataT *__FileStreamGetProbeData(CdxStreamT *stream)
{
//TODO  返回供 probe 的数据
}

static cdx_int32 __FileStreamRead(CdxStreamT *stream, cdx_void *buf, cdx_uint32 len)
{
//TODO  把从本地文件读取到的数据写到 buf 上
}

static cdx_int32 __FileStreamClose(CdxStreamT *stream)
{
//TODO  释放相应资源
}

static cdx_int32 __FileStreamGetIoState(CdxStreamT *stream)
{
//TODO  返回错误码
}

static cdx_uint32 __FileStreamAttribute(CdxStreamT *stream)
{
//返回属性标志
    return CDX_STREAM_FLAG_SEEK;
}

static cdx_int32 __FileStreamControl(CdxStreamT *stream, cdx_int32 cmd, cdx_void *param)
{
//暂时没有需要支持的命令，直接返回成功
    return CDX_SUCCESS;
}

static cdx_int32 __FileStreamSeek(CdxStreamT *stream, cdx_int64 offset, cdx_int32 whence)
{
//TODO  对本地文件 seek
}

static cdx_int64 __FileStreamTell(CdxStreamT *stream)
{
```

```
//TODO 获取本地文件的读写位置
}

static cdx_bool __FileStreamEos(CdxStreamT *stream)
{
//TODO  判断是否读到本地文件的尾部
}

static cdx_int64 __FileStreamSize(CdxStreamT *stream)
{
//TODO  获取文件大小
}

static cdx_int32 __FileStreamGetMetaData(CdxStreamT *stream, const cdx_char *key, cdx_void
**pVal)
{
//TODO  获取文件的元数据，如路径等。
}

static struct CdxStreamOpsS fileStreamOps =
{
        .connect = __FileStreamConnect,
        .getProbeData = __FileStreamGetProbeData,
        .read = __FileStreamRead,
        .write = NULL,
        .close = __FileStreamClose,
        .getIOState = __FileStreamGetIoState,
        .attribute = __FileStreamAttribute,
        .control = __FileStreamControl,
        .getMetaData = __FileStreamGetMetaData,
        .seek = __FileStreamSeek,
        .seekToTime = NULL,
        .eos = __FileStreamEos,
        .tell = __FileStreamTell,
        .size = __FileStreamSize,
};
static CdxStreamT *__FileStreamCreate(CdxDataSourceT *source)
{
//根据路径，创建本地文件句柄，初始化数据
}

CdxStreamCreatorT fileStreamCtor =
{
        .create = __FileStreamCreate
```

```
};
```

2）注册接口

```
static struct CdxStreamNodeS gStreamList[] =
{
    ……
    {"file", &fileStreamCtor},//注册 file stream，对应的 scheme 为"file"
    ……
};
```

3）使用该 stream

```
{
    CdxDataSourceT dataSource = {"file://xxxxxxx", NULL};
    CdxStreamT *sampleStream = CdxStreamOpen(CdxDataSourceT *source, pthread_mutex_t *mutex, cdx_bool *exit, CdxStreamT **stream, ContorlTask *streamTasks)

        //会根据 scheme 调用到 fileStreamCtor.create 即__FileStreamCreate 以及 connect
        //后面直接调用 CdxStreamRead、CdxStreamGetProbeData、CdxStreamClose 等接口对 sampleStream 做操作即可
}
```

7. 配置说明

（1）HTTP stream

关于 User-Agent：User-Agent（UA）是 HTTP 协议中的一部分，属于头域的组成部分，用于向服务端提供客户端所使用的浏览器类型、操作系统及版本等信息，在每次 HTTP 请求时发送到服务器。用户可以灵活设置 UA，通过 Android MediaPlayer 的 setDataSource 方法将所要的 UA 传递下来，如此 HTTP stream 便使用用户设定的 UA。若用户未设置，则默认使用"Allwinner/CedarX 2.7"。

8. Declaration

The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.