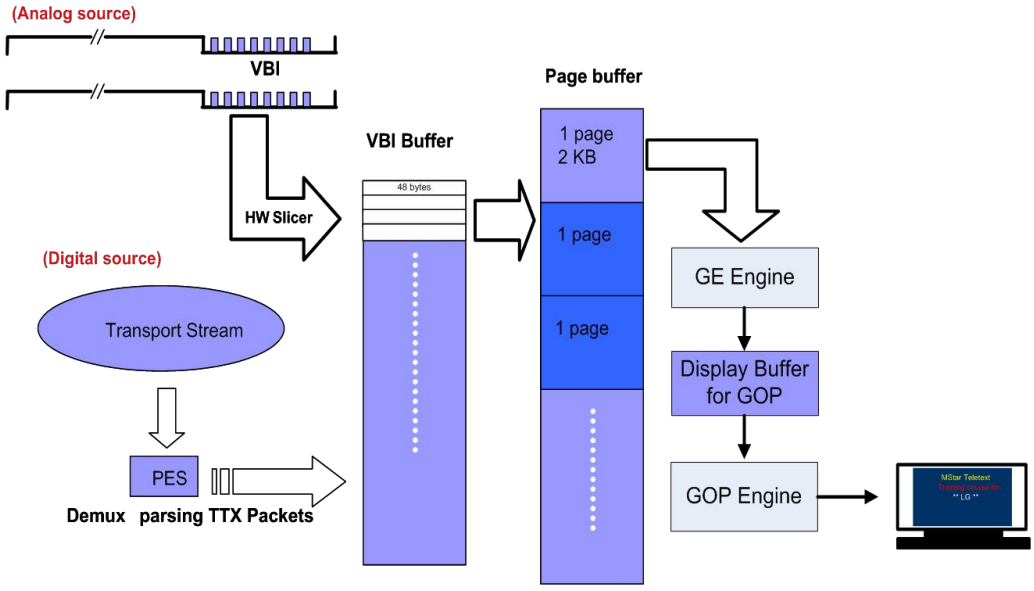
V-chip / CC /TTX/subtitle 功能：

图文

TTX(Teletext，图文)，分别有 Analog TTX（模拟图文，ATV 下）和 Digital TTX（数字图文， DTV 下），我们平时测试或者 debug 时，54200 可输出 Analog TTX，而用 码流机播放带 TTX 的码流测试可以输出 Digital TTX。
图文主要修改的部分是要确认图文按键功能是否正常，显示是否有异常。

TTX 基本架构

Analog TTX: TTX 数据嵌在Hsync line, 6~22 and 318~335
Digital TTX: TTX 数据被打包在码流里



TTX 4 显示模式
Normal: TTX 文本模式 (按 TTX 键)
Subtitle: 在视频上显示 TTX 字幕页面 (按 subcode 键)
Clock: 在视频上显示 clock,它会自动消失 (按 clock 键)
Status: 显示 20 status 以 packet8/30 format 1 显示字节 (东芝项目的要求,在换台时 显示)

```
8:
9:    switch(enTTXmode)
0:    {
1:       case TTX_MODE_NORMAL:
2:          {
3:             mapi_interface::Get_mapi_teletext()->Command(TTX_TEXT);
4:             bRet = TRUE;
5:          }
6:          break;
7:       case TTX_MODE_CLOCK:
8:          mapi_interface::Get_mapi_teletext()->Command(TTX_CLOCK);
9:          bRet = TRUE;
0:          break;
1:
2:       case TTX_MODE_SUBTITLE:
3:          if(_TTXInputType(m_enInputType) == EN_TTX_INPUT_ANALOG)
4:          {
5:             if(mapi_interface::Get_mapi_teletext()->GetSubtitlesAvailable())
6:             {
7:                //The initial state enter subtitle page mode
8:                m_u8SubtitlePageIndex = DEFAULT_SUBTITLE_INDEX;
9:                mapi_interface::Get_mapi_teletext()->GetSubtitlePage(m_u8SubtitlePageIndex,
0:                      &m_u16MagPage,
1:                      &m_u16SubPage);
2:                mapi_interface::Get_mapi_teletext()->CommandExt(TTX_SUBTITLE_TTX_ON,
3:                      TRUE,
4:                      m_u16MagPage,
5:                      m_u16SubPage);
6:                bRet = TRUE;
7:             }
8:             else
9:             {
0:                bRet = FALSE;
1:             }
2:          }
3:          else
4:          {
5:             // digital case
6:             mapi_interface::Get_mapi_teletext()->CommandExt(TTX_SUBTITLE_TTX_ON,
7:                   TRUE,
8:                   m_u16MagPage,
9:                   m_u16SubPage);
0:             bRet = TRUE;
1:          }
2:          break;
```

四种不同的模式会分别在按下不同的键的时候去响应，判断当前是否支持该模式，之后显示对应的功能，以下是一个打开 subtitles 的例子。图文中还有一类是 Subtitle mode，这个需要注意，在图文下有 subtitles 的页按下 subtitle 键会调出 subtitle 页，再按会切换。

```
switch (keyCodeToTrans) {
    case MKeyEvent.KEYCODE_SUBTITLE:
        if (TvCommonManager.getInstance().getCurrentTvSystem() == TvCommonManager.TV_SYSTEM_ATSC) {
            return false;
        }
        if (specificSourceIsInUse(from, TvCommonManager.INPUT_SOURCE_DTV)) {
            Intent intent = new Intent(from, SubtitleLanguageActivity.class);
            from.startActivity(intent);
            return true;
        } else {
            if (!getApp(from).isTTXEnable()) {
                Toast.makeText(from, "Not Support", Toast.LENGTH_SHORT);
                return true;
            }
            if (TvChannelManager.getInstance().isTeletextSubtitleChannel()) {
                if (TvChannelManager.getInstance().isTeletextDisplayed()) {
                    TvChannelManager.getInstance()
                            .sendTeletextCommand(TvChannelManager.TTX_COMMAND_SUBTITLE_NAVIGATION);
                } else {
                    TvChannelManager.getInstance()
                            .openTeletext(TvChannelManager.TTX_MODE_SUBTITLE_NAVIGATION);
                }
                return true;
            } else if(specificSourceIsInUse(from, TvCommonManager.INPUT_SOURCE_ATV)){
                errorinfo = "Subtitle"+" "+"UNSUPPORTED";
            }else {
                errorinfo = "Sorry, No Subtitle Available";
            }
            AlertDialog.Builder dialog = new AlertDialog.Builder(from);
            dialog.setTitle("Tip")
                    .setMessage(errorinfo)
                    .setPositiveButton("OK",
                            (dialog, arg1) -> {
```

TTX（图文）按键功能如下：

TEXT：进入/退出图文（有时与 MIX 复用，第一次显示图文，第二次 MIX 功能，图文图像同时显示，第三次退出图文）（退出图文方法：按 TEXT、SOURCE、EXIT 键）

INDEX：图文状态下，返回图文目录；DTV 非图文状态下，按此键可改变 PVR 列表节目的排列方式。

MIX：图文与图像同时显示。

REVEL：图文状态，显示隐藏的图文

1、ATV 图文 54200，102 页，按此键会显示“HELLO WORD”

2、播放 Dvbt(Sub&ttx).trp 码流在 P364,按此键显示或隐藏“413”

SUBPAGE：图文状态，显示当前页的子页，如在当前页 100，按下此键，显示“- + 100/----”可以直接输入子页码，可切换到子页

HOLD：锁定图文，使用图文停止刷新。

SIZE：图文状态，图文内容放大一倍显示，图文导航键（红、绿、黄、蓝）不作放大

CANCEL：图文状态，取消图文但不退出，只在屏幕左上角显示当前的页码，再按一次恢复正常显示。

Subtitle：

1.ATV 图文状态下，可直接跳到 SBUTITLE 页。接 54200 图文为开，当前未进入 TEXT 状态，按此键也应跳到 P150 页，此时按上下键不能翻页，再按一次退出 SUBTITLE 页（备注：此键与 SUBPAGE 键不复合情况下才能有此功能，若复合 SUBPAGE 键则在 ATV 下按此键优先实现 SUBPAGE 功能）

2.DTV 下无图文时按此键显示字幕

红、绿、黄、蓝键：对应图文的红、绿、黄、蓝快捷键，按下对应的颜色键后，将跳到与此颜色键对应的图文页下

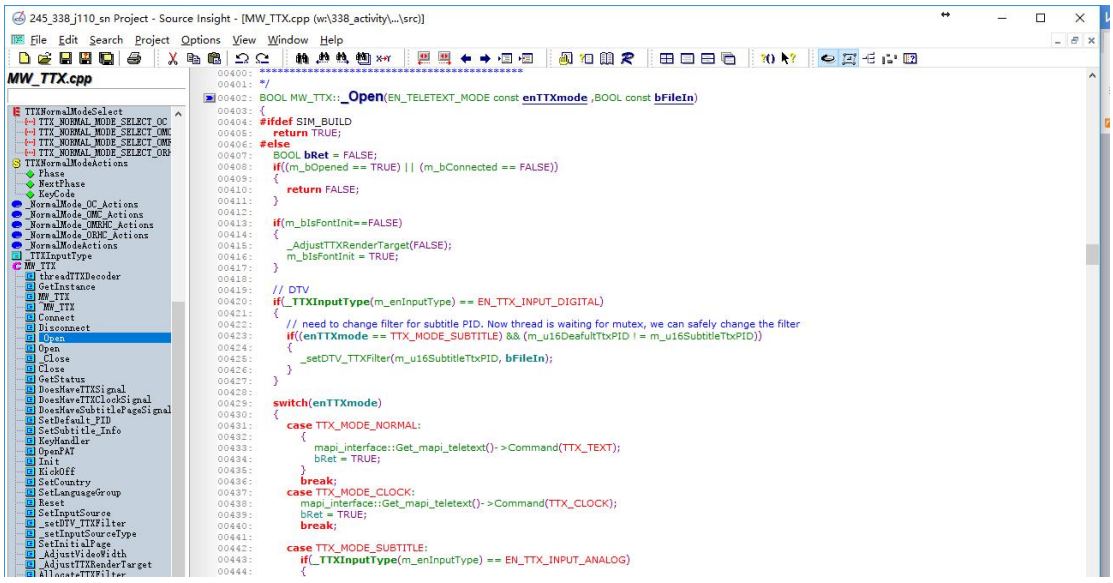
CHANNEL LIST：图文状态下，进入或退出 LIST，其它状态下无效.

客户一般有关图文的要求就是要修改遥控器图文的复用，这个只需要在 MTvPlayer 中的

TeletextActivity 中修改 onkeydown，增加对遥控器的判断之后增加对应按键的复用，其中 Index, Mix, List, Update, Reveal, Hold, Size, Channel up/down, 4 color keys, Subcode (subpage input)这些按键在 TeletextActivity 中响应，TTX，CLOCK,SUBTITLE 键在 SwitchPageHelper 中响应，如果是有关这几个键的功能问题，需要在这里修改。

```
}
if ("J110_RS41_T2".equals((SystemProperties.get("mstar.toptech.remote", "not-defined"))))
{
    switch (keyCode) {
        case MKeyEvent.KEYCODE_FAVORITE :
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_MIX);
            return true;
        case MKeyEvent.KEYCODE_SUBTITLE:
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_SIZE);//subittle-size
            Log.i("wsw", "TTX_COMMAND_SIZE");
            return true;
        case MKeyEvent.KEYCODE_MTS:
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_UPDATE);//audio-cancle
            Log.i("wsw", "TTX_COMMAND_UPDATE");
            return true;
        case KeyEvent.KEYCODE_MEDIA_REWIND:
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_INDEX);
            if (!mTvChannelManager.isTeletextDisplayed()) {
                finish();
            }
            return true;
        case KeyEvent.KEYCODE_MEDIA_FAST_FORWARD:
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_HOLD);
            return true;
        case KeyEvent.KEYCODE_MEDIA_PREVIOUS:
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_REVEAL);
            return true;
        case KeyEvent.KEYCODE_MEDIA_NEXT:
            mTvChannelManager.sendTeletextCommand(TvChannelManager.TTX_COMMAND_SUBPAGE);
            return true;

    }
```



Subtitle：

Subtitle 即是我们常说的字幕，常见于DVB 系统中。默认菜单中subtitle 为off，只有打 开subtitle 为 on，才可以显示出来。subtitle 总体分为 DVB subtitle 和 TTX subtitle。

1. SubtitleType

   DVB subtitle

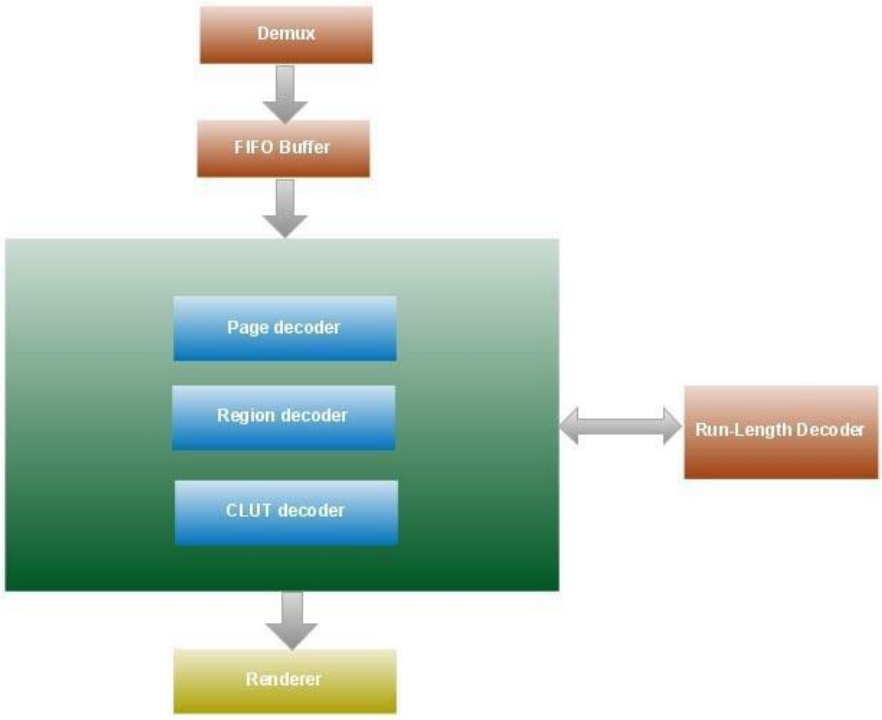
以 Bitmap 图形方式传输，系统通过 draw bitmap 实现

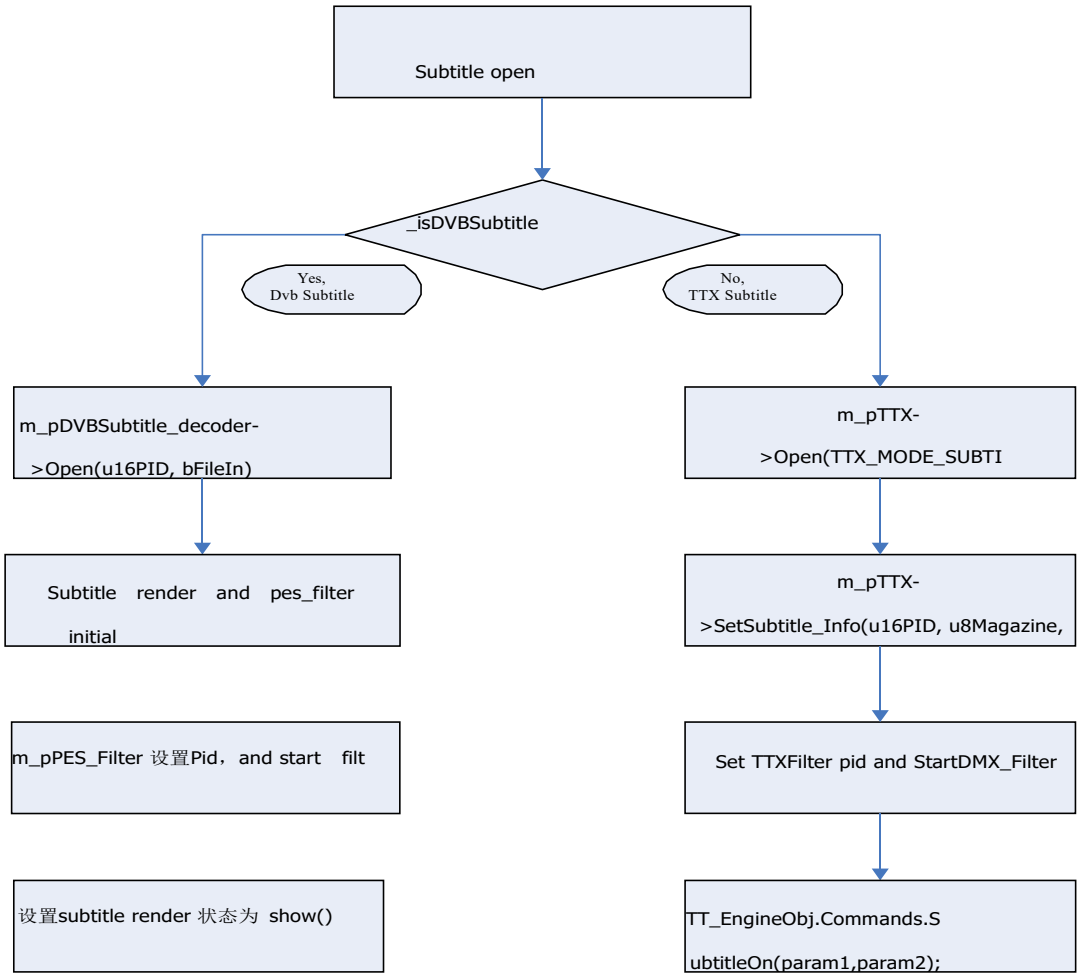
TTX subtitle

Stream 送过来的是 textual index，通过调用内建的字库来显示出来

常见的类型有：DVB_HD、DVB_HD_HOH、DVB_SD、TTX、TTX_HOH 一个节目可能同时携带多个 subtitle 类型，系统通过自动优先级算法来处理。 目前的 Priority 为:HOH＞Language＞DVB HD ＞ DVB SD ＞ TTX

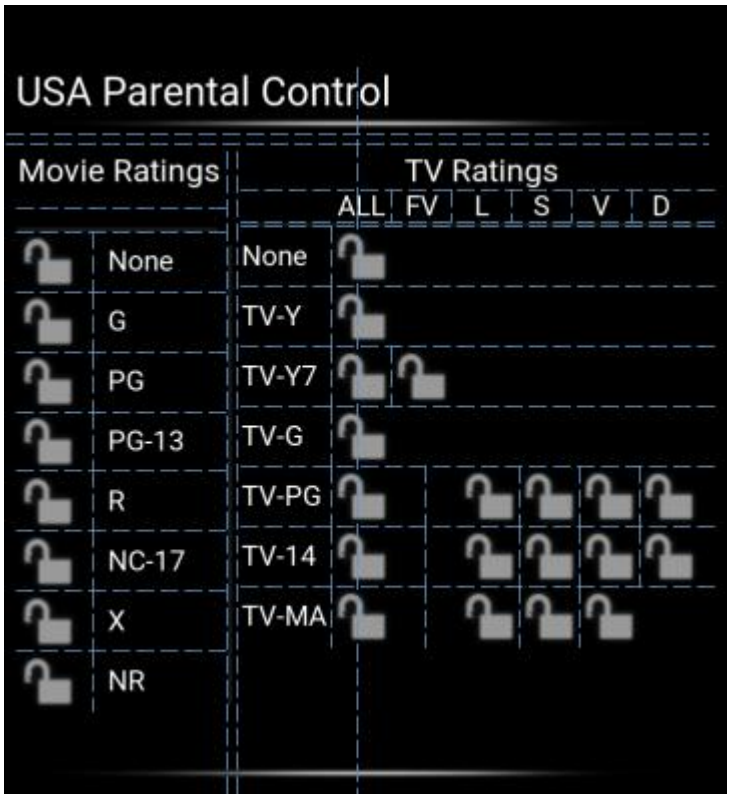
Subtitle Flow





V-chip：

V-Chip 为保护儿童免於受到电视上色情暴力节目的污染而实施、采用的制度。
出美国的电视都要求内置 V-chip 技术，家长已设置 Vchip 的等级，并设置密码，防止儿童取消限制观看色情或暴力等节目。美国的电视等级主要分为以下几个级别：
美国电视节目内容分级：
主要分为电视和电影等级，电影分为 G；PG;PG13;R;NC-17;X,NR.
电视等级分为 Y;Y7;G;PG;14;MA;其中，不同的等级锁主不同等级的内容，又分为 FV;L;S;V，D,可以选择对不同的情形进行锁住。
是内容的分级制度，主要用于 ATSC 制式，美国和加拿大地区控制电视内容的分级



加拿大的又分为英语的标题和法语的：等级为：E;C;C8+;G;PG;14+;18+;
法语的级别为：E;G;8 ans+;13 ans+;16 ans+,18 ans+.

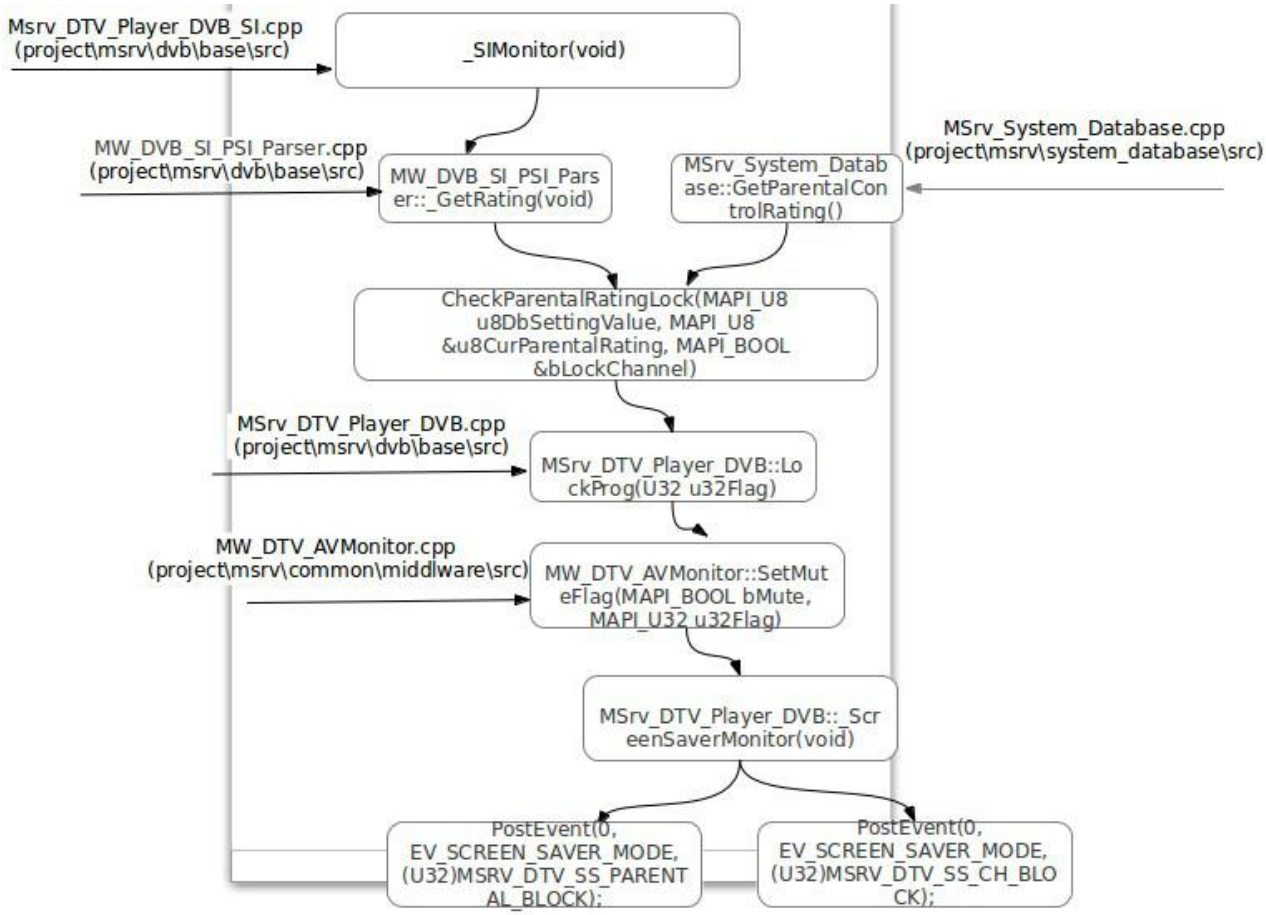
V-chip 有关的 API 介绍：

V-chip 在 MW 层有自己的一个 Thread 来监听从码流里面拿到的 V-chip 数据，这些数据可以存放在 DTV 的 SI 中，也可以存放在 ATV/AV 的 CC 中，V-chip 线程实现函数：
void MW_VCHIP::MonitorVchip(void)
在切换到带 Vchip 功能的 Source 时会执行 V-chip connect：
BOOL MSrv_Control_ATSC::VChipConnect()，在 source 被切走时会执行
BOOL MSrv_Control_ATSC::VChipDisconnect()。
V-chip 作为 TV 的 Lock system 的设定，它的接口分为两类，一种是对设定提供数据库支持，以实现断电保存，另外一种是提供具体设定的接口用来实现功能，UI 去设定一个 V-chip table 锁定的等级时会执行两个步骤，一是设定具体实现的 API，其二就要设定数据库接口。那么下面就分两类介绍 V-chip 有关的 API。
V-chip 设定数据库的接口：
void MSrv_System_Database_ATSC::SetVchipSetting(ST_VCHIP_SETTING *pValue)
其中结构体 ST_VCHIP_SETTING 包含了各种 VChip Rating 标准。
V-chip 设定实现的接口：
BOOL MSrv_Control_ATSC::SetVChipGuideline(U8 RatingType, U8 Para1, U8 Para2, U8 Para3) 第一个参数 Rating type 表示当前设定的是哪个 Rating 标准，每个 rating 标准其实是一个表格，简单的如 Canada English/French 就只用一维表格就能表示，因此当 RatingType=2/3 时只用 Para1 参数；TV Rating/MPAA 标准需要用到二维表格；RRT5 需要用到三维表格，详情请

见 Spec CEA-766。

Rating Lock 是指父母根据小孩的年纪设定一个等级M，如果码流中所携带的等级N高于父母所设定的等级M，将会上锁即屏幕黑屏弹出密码框。需要输入密码才能观看。主要是 针对 DTV,因 ATV 等 Source 没有 Rating Lock.

Rating lock 的实现流程：



Rating lock 的主要功能都是通过 ParentalControlManagerservice.cpp 来实现。首先当要使用等级锁或者是频道锁时，都需要先将父母锁的总开关打开。通过 call void setSystemLock(boolean islocked),然后通过 setParentalControlRating(int rating) 设置等级， 当解析到码流里携带有 rate 信息时，会将改 rate 与 db 里面 set 的 rate 对比，如果大于将 mute screen,通过上抛 event 给上次 an,从而弹出密码框。

那么 rate lock 又是怎么解锁呢？这里的解锁实质就是解 mute，首先保证所输入的 password 正确，然后直接 call BOOL Msrv_DTV_Player_DVB::UnlockChannel(void),发送消息 SetCMD(MSRV_DTV_DVB_CMD_UNLOCK_CHANNEL, 0, 0);当接收到这个消息后就会去解 mute,并且抛 event 给ＵＩ，dismiss 掉密码框。

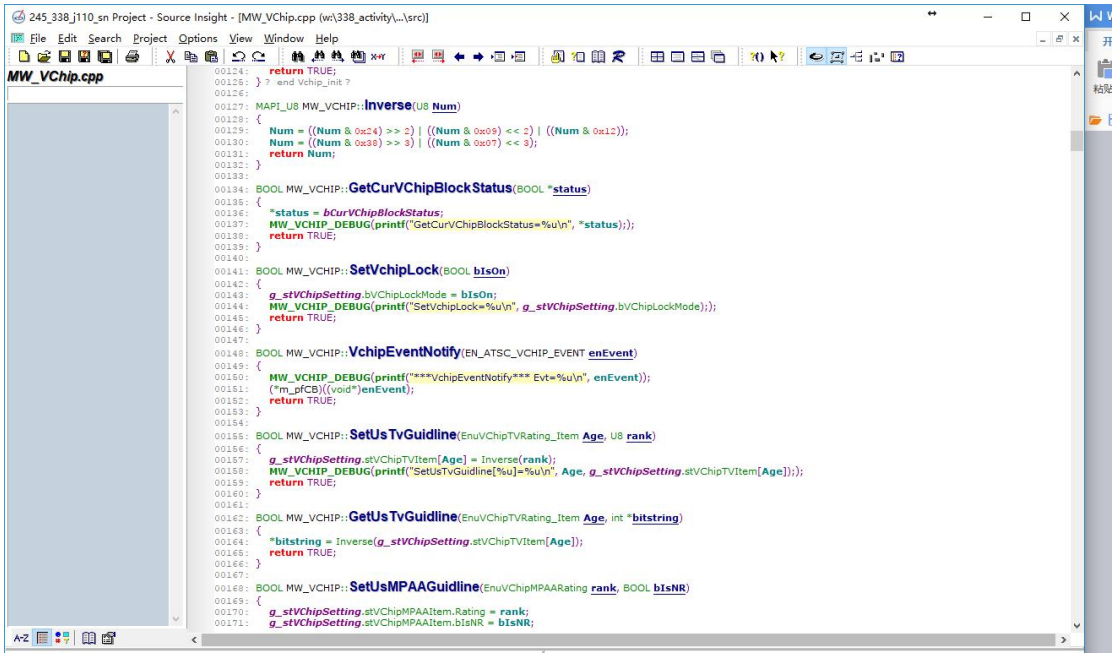
```
887: //-------------------------------------------------------------------------------------------
888: BOOL MSrv_DTV_Player_DVB::UnlockChannel(void)
889: {
890:     MSRV_DVB_PLAYER_FUNCTION("MSrv_DTV_Player_DVB::UnlockChannel()\n");
891:
892:     CHECK_DVB_PLAYER_INITIAL_DONE();
893:
894:     return SetCMD(MSRV_DTV_DVB_CMD_UNLOCK_CHANNEL, 0, 0);
895: }
896:
```

如何通过码流解析软件查看码流里面rate 值？
在我们 debug 问题中往往会出现unlock 的问题，这时候就需要借助码流分析软件查看码流 rate 信息是否大于所设定的值。（有时候存在一条码流有多个 rate 值） 用码流分析工具check 出这条码流的哪个servie 带parental guide 信息，解读这个信息，
->从码流工具解出来：

```
EIT Table ID 0x4e Service ID 371 Version 31 Section 0 [PID 18 (0x0012)] (LM TV Sarthe {actual} {present/following})
EIT Table ID 0x4e Service ID 371 Version 31 Section 1 [PID 18 (0x0012)] (LM TV Sarthe {actual} {present/following})
```

这两个event 是有加父母锁：

```
short_event_descriptor
parental_rating_descriptor
    descriptor_tag = 85 (0x55)
    descriptor_length = 4 bytes
    parental_rating_descriptor = FRA 7
```



CC

Closed Caption，简称 CC，意思就是隐藏的带有解释意味的字幕（CAPTION）。CAPTION 与我们常见的一般字幕（SUBTITLE）的用法是有区别的，它是在无音状态下通过进行一些解释性的语言来描述当前画面中所发生的事情的字幕，例如画面中出现了背景的声音的时候，CAPTION 都会通过字幕进行提示。主要用于ATSC 机型（美国，加拿大等国家），ISDB 机型（巴西等国家）。

一般这类型的字幕都是为听力有障碍或者无音条件下观看节目的观众所准备的，在美国执行的一些非强制性的标准中要求一般的电视及录像节目都要为这样的观众提供 CAPTION 的字幕，如果不做直接的可显示字幕的话，也要求将其做成可隐藏的字幕，也就是 CLOSED CAPTION，即 CC 字幕；

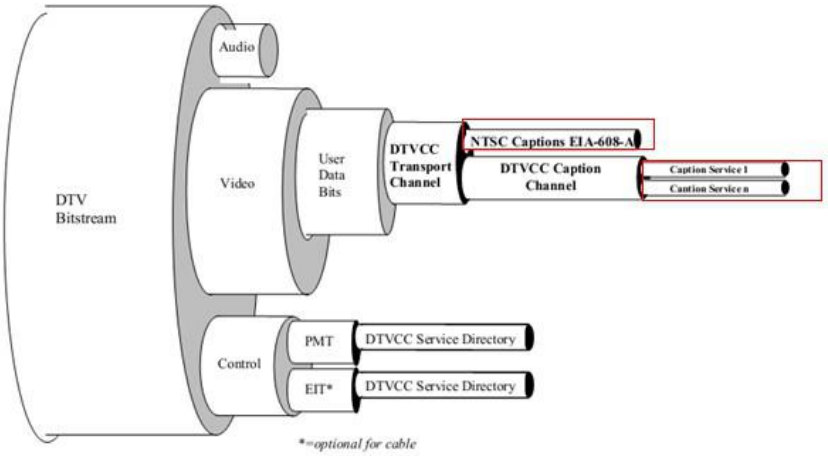
CCD 的数据主要来源于：NTSC (EIA-608B)(模拟数据)& ATSC (EIA-708B)(数字数据) CC Data。ATV 下只有 模拟的数据。DTV 下同时存在模拟和数字的数据。

608 包括：CC1，CC2，CC3，CC4，TEXT1，TEXT2，TEXT3，TEXT4，XDS（Extended Data services）。CC1，CC2，CC3，CC4 可以用来传输不同的语言文字，内容主要是图像中人物的 对白，在使用时将相应的文字显示在说话人嘴巴的附近；TEXT1，TEXT2，TEXT3，TEXT4 主 要用于传输一些信息，如天气预报，新闻等等；XDS 传输的数据主要用于 V-CHIP（Program Blocking）使用。608 是从 VBI 里面拿到的数据。

708 包括：

EIA-708-B

- Consisting DTVCC & NTSC CC Data (EIA-608B)
- High Data Rate: Pre-Allocated Bandwidth(9600 bps, 2400 bps for each service)
- Caption Display: Window Concept (more attribute: size, color, font)



```
                return true;
        case MKeyEvent.KEYCODE_CC:
            boolean bIsCcSupported = false;
            if (true == Utility.isATSC()) {
                if (TvCommonManager.getInstance().isSupportModule(
                        TvCommonManager.MODULE_ATSC_CC_ENABLE)
                        && TvCommonManager.getInstance().isSupportModule(
                                TvCommonManager.MODULE_NTSC_CC_ENABLE)) {
                    // ATSC TV System using [DTV]MODULE_ATSC_CC_ENABLE +
                    // [ATV]MODULE_NTSC_CC_ENABLE
                    bIsCcSupported = true;
                }
            } else if (true == Utility.isISDB()) {
                if (mTvCommonManager.getCurrentTvInputSource() == TvCommonManager.INPUT_SOURCE_DTV)
                    return true;
                if (TvCommonManager.getInstance().isSupportModule(
                        TvCommonManager.MODULE_ISDB_CC_ENABLE)
                        && TvCommonManager.getInstance().isSupportModule(
                                TvCommonManager.MODULE_NTSC_CC_ENABLE)) {
                    // ISDB
                    // [ATV]MODULE_NTSC_CC_ENABLE
                    bIsCcSupported = true;
                }
            } else {
                if (TvCommonManager.getInstance().isSupportModule(
                    TvCommonManager.MODULE_NTSC_CC_ENABLE)&& (mTvCommonManager.getCurrentTvInputSource() != TvCommonMa
                    // ASIA CC = [ATV]MODULE_NTSC_CC_ENABLE
```

Cannot resolve symbol 'TvCommonManager'

CC 在 DTV 中有分为 ATSC 的 CC 和 ISDB 的 CC ，针对不同的制式，cc 的模式也不同：

ATSC CC:

```
    */
    @Override
    public int getNextClosedCaptionMode() throws RemoteException {
        int closedCaptionMode = getClosedCaptionMode(); // Get Current ClosedCaption Mode
        int nextClosedCaptionMode = closedCaptionMode;
        if (TvCommonManager.getInstance().isSupportModule(TvCommonManager.MODULE_CC)
            || (TvCommonManager.getInstance().isSupportModule(TvCommonManager.MODULE_ATSC_CC_ENABLE)
            && TvCommonManager.getInstance().isSupportModule(TvCommonManager.MODULE_NTSC_CC_ENABLE))) {
            // FOR ATSC (ATSC + NTSC)
            // MODULE_CC = MODULE_ATSC_CC_ENABLE + MODULE_NTSC_CC_ENABLE

            // In Supernova
            // ATSC_CC_MODE_ON = 0
            // ATSC_CC_MODE_OFF = 1
            // ATSC_CC_MODE_ONMUTE = 2
            switch (closedCaptionMode) {
                case CLOSED_CAPTION_OFF:
                    nextClosedCaptionMode = CLOSED_CAPTION_ON_WHEN_MUTE;
                    break;
                case CLOSED_CAPTION_ON:
                    nextClosedCaptionMode = CLOSED_CAPTION_OFF;
                    break;
                case CLOSED_CAPTION_ON_WHEN_MUTE:
                    nextClosedCaptionMode = CLOSED_CAPTION_ON;
                    break;
                default:
                    Log.d(TAG, "Not a valid ClosedCaption Mode in MODULE_CC");
                    break;
            }
```

ISDB CC :

```
    } ? end if TvCommonManager.getIn... ?  else if ((TvCommonManager.getInstance().isSupportModule(TvCommonManager.MODULE_ISDB_CC_ENABLE)
            && TvCommonManager.getInstance().isSupportModule(TvCommonManager.MODULE_NTSC_CC_ENABLE))
        || TvCommonManager.getInstance().isSupportModule(TvCommonManager.MODULE_BRAZIL_CC)) {
        // ISDB => MODULE_BRAZIL_CC = MODULE_ISDB_CC_ENABLE + MODULE_NTSC_CC_ENABLE
        switch (closedCaptionMode) {
            case CLOSED_CAPTION_OFF:
                nextClosedCaptionMode = CLOSED_CAPTION_ON;
                break;
            case CLOSED_CAPTION_ON:
                nextClosedCaptionMode = CLOSED_CAPTION_CC1;
                break;
            case CLOSED_CAPTION_CC1:
                nextClosedCaptionMode = CLOSED_CAPTION_CC2;
                break;
            case CLOSED_CAPTION_CC2:
                nextClosedCaptionMode = CLOSED_CAPTION_CC3;
                break;
            case CLOSED_CAPTION_CC3:
                nextClosedCaptionMode = CLOSED_CAPTION_CC4;
                break;
            case CLOSED_CAPTION_CC4:
                nextClosedCaptionMode = CLOSED_CAPTION_TEXT1;
                break;
            case CLOSED_CAPTION_TEXT1:
                nextClosedCaptionMode = CLOSED_CAPTION_TEXT2;
                break;
            case CLOSED_CAPTION_TEXT2:
                nextClosedCaptionMode = CLOSED_CAPTION_TEXT3;
                break;
            case CLOSED_CAPTION_TEXT3:
                nextClosedCaptionMode = CLOSED_CAPTION_TEXT4;
                break;
            case CLOSED_CAPTION_TEXT4:
                nextClosedCaptionMode = CLOSED_CAPTION_OFF;
                break;
            default:
                Log.d(TAG, "Not a valid ClosedCaption Mode in MODULE_BRAZIL_CC");
                break;
        } ? end switch closedCaptionMode ?
        int currentInputSrc = TvCommonManager.getInstance().getCurrentTvInputSource();
```


```
00208:
00209:    m_pCcDispWin = NULL;
00210:    m_pCcPrimary = NULL;
00211:
00212:    m_u32PreviewCheckTime = 0;
00213:    m_bIsFlash = FALSE;
00214:    m_enARGBMode = CAPTION_COLORMODE_ARGB8888; //Bingo 20170103 modify for CC issue,mantis:0024020
00215:
00216:    m_pDispWin = m_pBackWin = m_pBackBlinkWin = NULL;
00217:    m_capPrimary = m_previewSur = m_previewBlinkSur = NULL;
00218:
00219:
00220:    m_wInfo.x = 0;
00221:    m_wInfo.y = 0;
00222:    m_wInfo.w = 0;
00223:    m_wInfo.h = 0;
00224:
00225:    m_bThreadActive = FALSE;
00226:    m_bThread_EnDecode = FALSE;
00227:    m_bSignalstable = FALSE;
00228:    m_bVchipBlockStatus = FALSE;
00229:    CC_thread_result = PTHREAD_CANCELED;
00230:    m_bRvuDssType = FALSE;
00231:    pthread_mutex_init(&m_CCMutex, NULL);
00232:
00233:    CC_DBG("\n...[%s, %d] construct...\n", __FUNCTION__, __LINE__);
00234:
00235:    for(U8 i = 0; i < CC_FONT_TABLE_NUMBER; i++)
00236:    {
00237:        m_ccFont_t[i] = NULL;
00238:        m_ccFontGlyph_t[i] = NULL;
00239:    }
00240:
00241:    memset(&m_thread_id, 0, sizeof(pthread_t));
00242: } ? end MW_CC ?
00243:
00244: MW_CC::~MW_CC(void)
00245: {
00246:    pthread_mutex_destroy(&m_CCMutex);
00247:    CC_DBG("\n...[%s, %d] destruct...\n", __FUNCTION__, __LINE__);
00248: }
00249:
00250: /*
00251:  ************************************************
00252:  FUNCTION   : threadCCAcquisition
00253:  USAGE      : Create after MW_CC connect
00254:               Exit when disconnect.
00255:  INPUT      : None
00256:  OUTPUT     : None
00257:  ************************************************
00258: */
```