视频编解码和摄像头应用案例

本文档涉及案例包括：

- 打开摄像头并编码存储摄像头视频，然后解码播放视频
- 编码视频流并通过网络传输，接收端解码并播放
- 同时解码并播放两路视频
- 在视频播放窗口上叠加图标

1. 在动手实践本应用案例前先预定 i.MX8MP 开发板：

   1）打开云实验室网页，点击右上角登陆按钮输入账号密码。

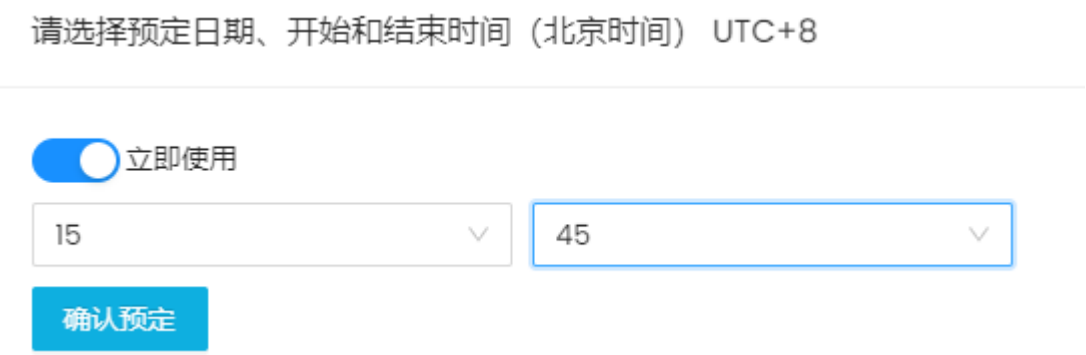
   https://aiotcloud.nxp.com.cn/

   登陆后依次点击硬件 -> i.MX 8 系列开发板

   找到 i.MX 8M Plus “可立即使用” 状态的板子，点击“8MPLUSLPD4-PEVK“进入。

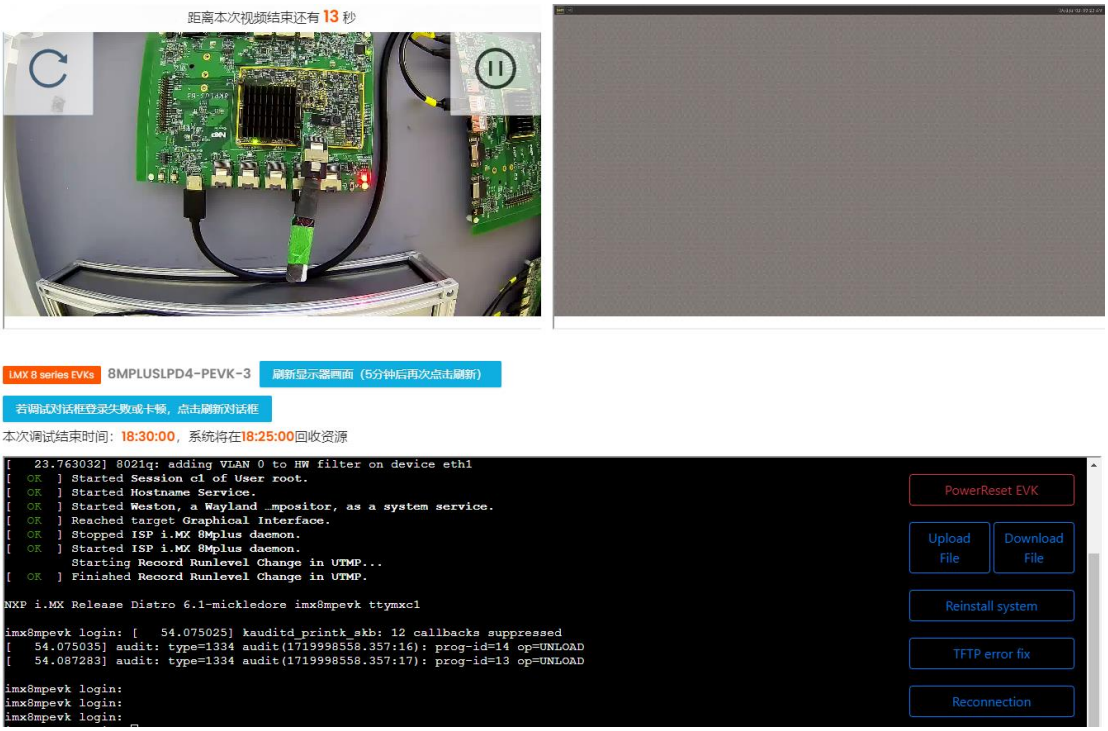
   然后点击“立即预定“按钮：

   选择立即使用，填写使用结束时间



   然后点击进入我的预定，

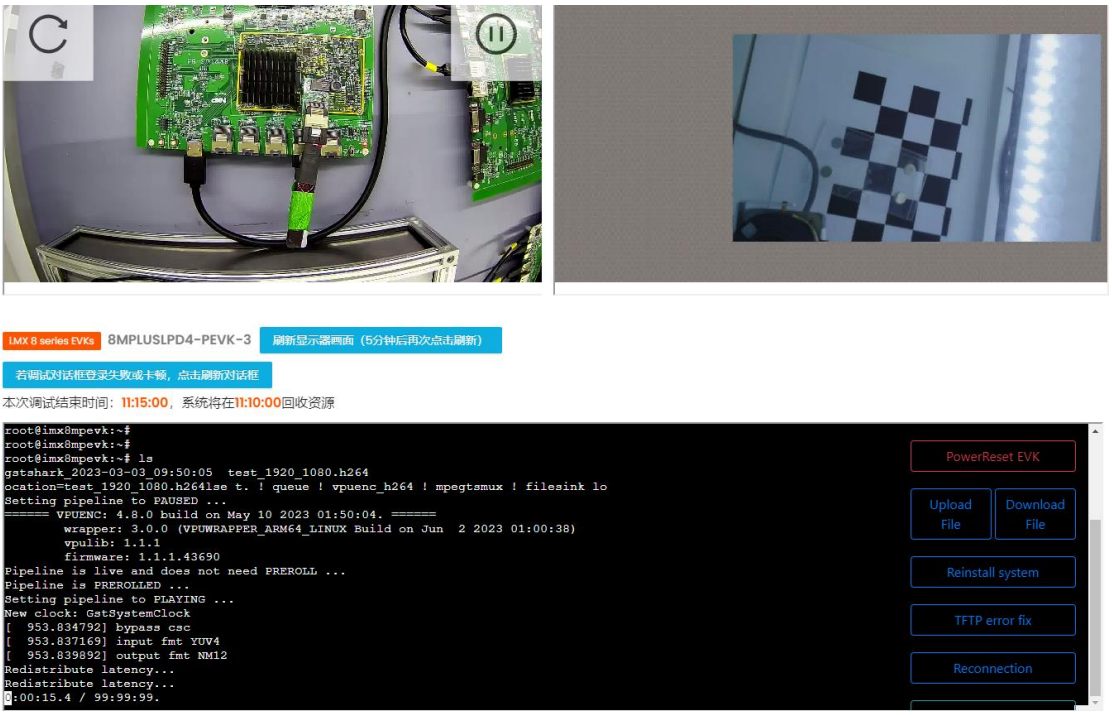
点击右侧蓝色“调试”按钮，之后就进入到板子的实物页面和系统启动 log 页面。到此为止，板子预定并且启动成功。



2. 使用 gplay 播放，gplay 是 gstreamer 的前端，实现音视频的播放，它的底层会根据硬件选择合适的插件来做硬件加速。
   首先执行程序，利用板子摄像头拍摄一段视频并压缩存储到本地，作为实验用。

```
gst-launch-1.0 v4l2src device=/dev/video3 ! video/x-raw,format=NV12,width=1920,height=1080 ! tee name=t ! queue ! waylandsink window-width=1280 window-height=720 sync=false t. ! queue ! vpuenc_h264 ! mpegtsmux ! filesink location=test_1920_1080.h264
```

   等待约 30s。

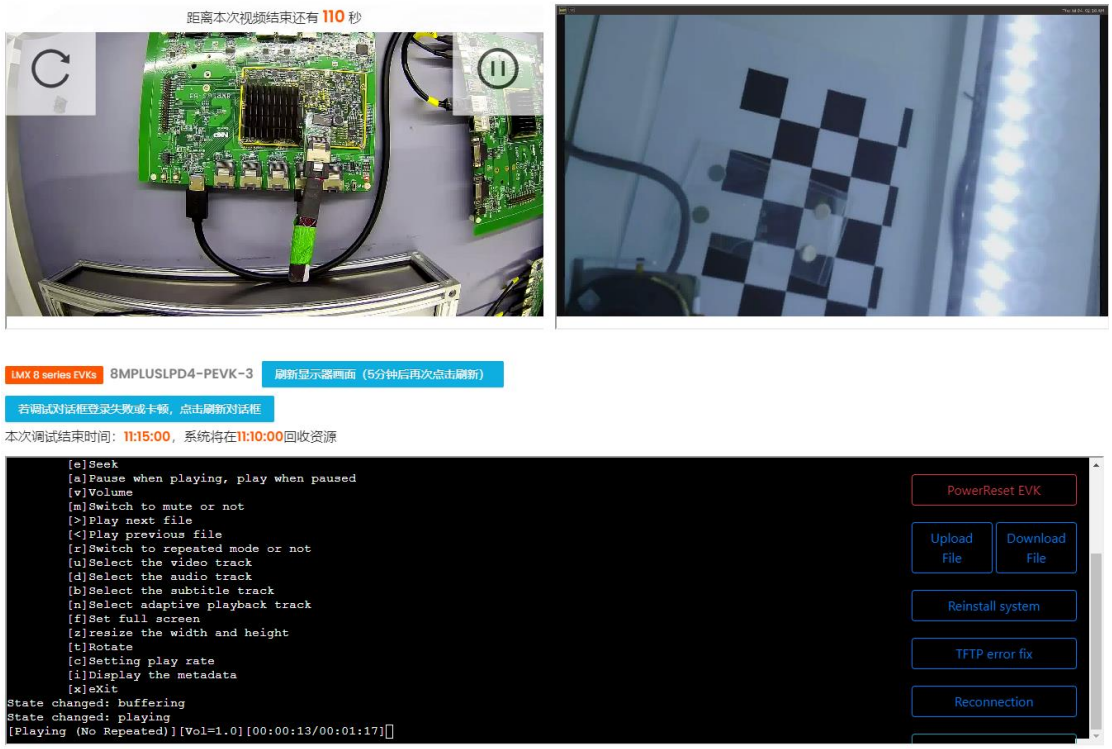


这样本地就有了视频源文件 test_1920_1080.h264。板子执行 ls 命令即可看到：

```
root@imx8mpevk:~# ls
gstshark_2023-03-03_09:50:05  test_1920_1080.h264
```

然后使用 gplay 播放即可：

```
root@imx8mpevk:~#gplay-1.0 test_1920_1080.h264
```



3. i.MX 板子编码本地摄像头视频流并通过网络传输，远端 i.MX 板子接收网络视频流并解码，以 i.MX8MP 为例，使用到多个硬件模块，有摄像头 MIPI CSI 接口，ISI，网络，

VPU 编解码，显示控制器。

这里使用本地网络 loopback（从本地网口发送再回到本地网口接收）的方式来模拟网络发送接收。

重启板子，等板子启动后输入 root 登录，板子上执行：

```
~#gst-launch-1.0 v4l2src device=/dev/video3 io-mode=dmabuf  ! video/x-raw,format=NV12,width=1280,height=720 ! v4l2h264enc output-io-mode=dmabuf-import ! rtph264pay ! udpsink host=127.0.0.1 port=6012 &
```

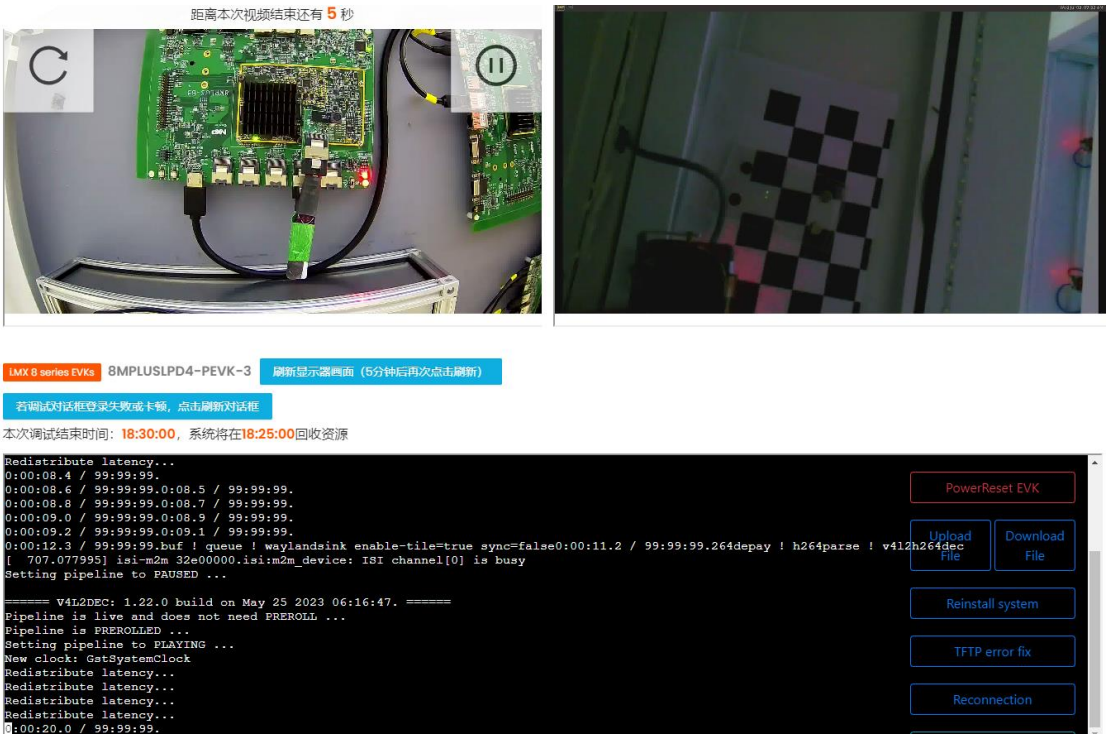
```
====== V4L2ENC: 1.22.5 build on Oct 19 2023 01:57:00. ======
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Redistribute latency...
[  917.806194] 1500000001 fail to communicate with daemon, error=-512, cmd=3
[  929.403727] bypass csc
[  929.403734] input fmt YUV4
[  929.403739] output fmt NM12
Redistribute latency...
0:00:08.7 / 99:99:99.
```

对端网络接收、解码并显示，因为是用同一个板子模拟网络发送，所以板子上执行：

```
~#gst-launch-1.0 udpsrc address=127.0.0.1 port=6012 ! application/x-rtp,encoding-name=H264 ! rtph264depay ! h264parse ! v4l2h264dec capture-io-mode=dmabuf ! queue ! waylandsink enable-tile=true sync=false
```

```
====== V4L2ENC: 1.22.5 build on Oct 19 2023 01:57:00. ======
Pipeline is live and does not need PREROLL ...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
New clock: GstSystemClock
Redistribute latency...
[  917.806194] 1500000001 fail to communicate with daemon, error=-512, cmd=3
[  929.403727] bypass csc
[  929.403734] input fmt YUV4
[  929.403739] output fmt NM12
Redistribute latency...
0:00:08.7 / 99:99:99.
```

可以看到屏幕显示了网络发送的视频流：



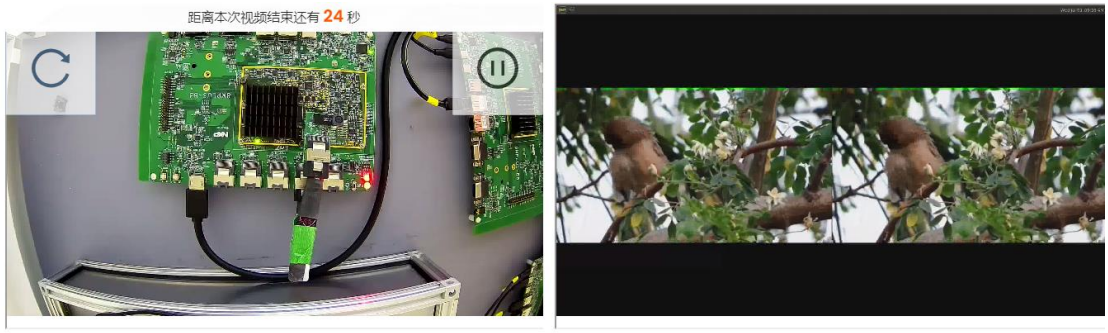
4. 同时播放两个视频，使用的硬件加速有硬件 VPU 解码和 GPU 进行窗口合成，在测试这个功能前需要重启板子，点击 log 窗口右上角的 PowerReset EVK 键，

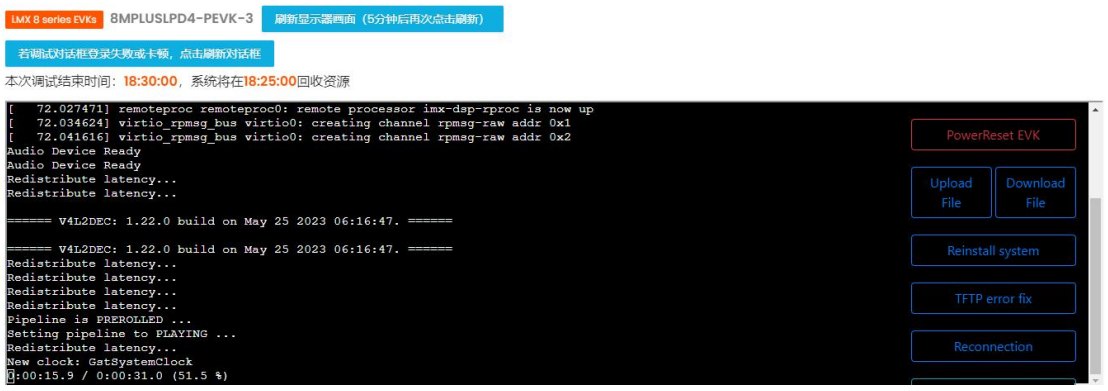
等待重启完毕，输入 root 命令登录板子。
然后执行:

```
gst-launch-1.0 imxcompositor_g2d background=0x00FFFFFF name=comp
sink_0::xpos=0 sink_0::ypos=0 sink_0::width=960 sink_0::height=540 sink_1::xpos=960
sink_1::ypos=0 sink_1::width=960 sink_1::height=540 ! waylandsink uridecodebin
uri=file:///home/root/BirdNoSound.mp4 ! queue ! comp.sink_0 uridecodebin
uri=file:///home/root/BirdNoSound.mp4 ! queue ! comp.sink_1
```

```
====== BEEP: 4.8.2 build on Oct 10 2023 02:31:27. ======
        Core: DSP decoder Wrapper  build on Jun  5 2023 05:45:53
 file: /usr/lib/imx-mm/audio-codec/wrap/lib_dsp_wrap_arm_elinux.so

====== BEEP: 4.8.2 build on Oct 10 2023 02:31:27. ======
        Core: DSP decoder Wrapper  build on Jun  5 2023 05:45:53
 file: /usr/lib/imx-mm/audio-codec/wrap/lib_dsp_wrap_arm_elinux.so
Audio Device Ready
Audio Device Ready
Redistribute latency...
Redistribute latency...

====== V4L2DEC: 1.22.5 build on Oct 19 2023 01:57:00. ======

====== V4L2DEC: 1.22.5 build on Oct 19 2023 01:57:00. ======
Redistribute latency...
Redistribute latency...
Pipeline is PREROLLED ...
Setting pipeline to PLAYING ...
Redistribute latency...
New clock: GstSystemClock
0:00:06.9 / 0:00:31.0 (22.3 %)
```

执行效果的画面如下：





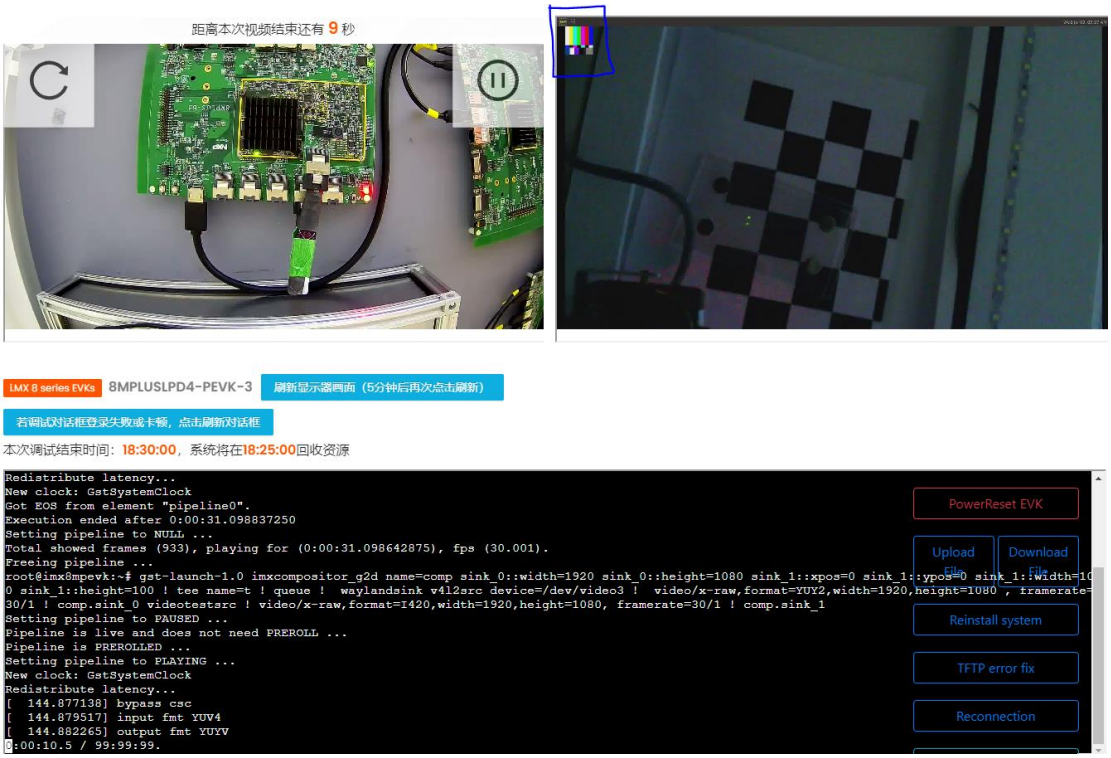
等待视频播放完毕或者按 Ctrl+C 键停止程序

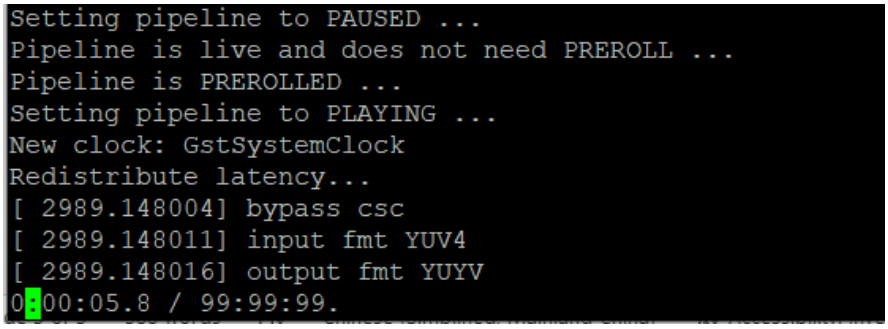
5. 在播放视频的窗口上叠加图标或者按钮等，板子上执行：

```
gst-launch-1.0 imxcompositor_g2d name=comp sink_0::width=1920
sink_0::height=1080 sink_1::xpos=0 sink_1::ypos=0 sink_1::width=100
sink_1::height=100 ! tee name=t ! queue !  waylandsink v4l2src
```

```
device=/dev/video3！ video/x-raw,format=YUY2,width=1920,height=1080 ,
framerate=30/1 ! comp.sink_0 videotestsrc ! video/x-
raw,format=I420,width=1920,height=1080, framerate=30/1 ! comp.sink_1
```

执行效果如下：





按 Ctrl + c 关闭程序的运行。