Play Video

Introduction to the results of routine experiments

In this section, we will learn how to use K230 to play MP4 format videos.

The example code of this section is located in: [Source code/10.Media/04.play_video.py]

We use CanMV IDE to open the sample code and connect K230 to the computer via USB.

Click the Run button in the lower left corner of CanMV IDE, and K230 will start playing the video (the default video is [/data/video/video.mp4])

> Note 1: The current version of K230 can only play videos with audio format g711. If the video cannot be played, it is likely that the audio in your video file is in the mainstream ACC format.
>
> Note 2: The video recorded using the example code in the previous section can be played
>
> Note 3: The sound of the currently recorded video will be intermittent when playing. This problem will be fixed in subsequent versions.

Code Analysis

Let's analyze the structure of this MP4 player code:

1. Overall architecture

- It mainly contains two functions:
  - `player_event`: Player event callback processing
  - `play_mp4_test`: Core MP4 playback function
- Global state management: using `start_play` variables to track playback state

2. Player initialization and configuration

```
player = Player(Display.ST7701)  # 使用ST7701 LCD屏作为输出显示，最大分辨率640*480 Use the ST7701 LCD screen as the output display, with a maximum resolution of 640*480
player = Player(Display.ST7701)  # 使用ST7701 LCD屏作为输出显示，最大分辨率640*480 load mp4 file
player.set_event_callback(player_event)  # 设置播放器事件回调函数 Set the player event callback function
```

3. Event handling mechanism

```
def player_event(event, data):
    global start_play
    if(event == K_PLAYER_EVENT_EOF):  # 检测到播放结束事件（End Of File） Detect the
end of playback event (End Of File)
        start_play = False  # 设置播放结束标识 Set the playback end mark
```

4. Playback process control

```
player.start()  # 开始播放视频 Start playing the video
start_play = True  # 设置播放状态为开始播放 Set the playback status to start playing
try:
    while(start_play):
        time.sleep(0.1)  # 休眠0.1秒，减少CPU占用 Sleep for 0.1 seconds to reduce
CPU usage
        os.exitpoint()  # 处理退出点，允许程序正常退出 Process the exit point,
allowing the program to exit normally
```

5. Error handling mechanism

```
try:
    # 播放循环 Play loop
except KeyboardInterrupt as e:  # 捕获键盘中断异常 Capture keyboard interrupt
exception
    print("user stop: ", e)
except BaseException as e:  # 捕获所有其他异常 catch all other exceptions
    import sys
    sys.print_exception(e)
```

6. Display device selection

```
# player=Player(Display.VIRT)  # 使用IDE作为输出显示，可以设定任意分辨率 Use IDE as the
output display, you can set any resolution
player=Player(Display.ST7701)  # 使用ST7701 LCD屏作为输出显示，最大分辨率640*480 Use
the ST7701 LCD screen as the output display, with a maximum resolution of
640*480
```