

数字系统II 实验 报告四

Copyright (c) 2019 Minaduki Shigure.

南京大学 电子科学与工程学院 吴康正 171180571

项目repo地址: https://git.nju.edu.cn/Minaduki/beaglebone_proj

实验目的

1. 了解软件层次结构，学习解决依赖关系。
2. 在提供的嵌入式平台上通过源码移植一个应用软件。
3. 通过USB声卡和移植的MPlayer播放音频文件。
4. 通过HDMI显示器播放简单的视频文件。

实验环境

1. 硬件环境

实验使用了TI的BeagleBone Black开发板作为实验环境，其参数如下：

- 处理器：基于ARM Cortex-A8架构的TI AM3359 Sitara @ 1GHz
- 内存：板载512MiB DDR3
- 存储：板载4GiB 8-bit eMMC闪存
- 拓展存储：支持micro SD存储卡
- 网络界面：RJ-45接口百兆以太网
- 数字多媒体输出：micro HDMI接口
- 拓展接口：UART、GPIO、SPI、I2C等
- USB声卡

使用的显示器：

- 分辨率：最大1920*1080
- 色彩空间：RGB565

2. 软件环境

- 上位机：使用Ubuntu 19.10系统的x86 PC
- Linux源码：[版本4.4.155](#)
- Busybox源码：[版本1.30.1](#)
- 编译器：The GNU Compiler Collection 9.2.1
- bootloader：U-boot
- zlib源码：版本1.2.11
- libmad源码：版本0.15.1b
- ffmpeg源码：版本4.2
- mplayer源码：版本1.4

3. 网络环境

- 网关: 192.168.208.254
- 上位机: 192.168.208.35
- 开发板: 192.168.208.121

实验原理

关于实验原理:

具体的实验原理中，实验指导书有所提及的在这里不再重复叙述。这里仅进行一些补充说明。

依赖关系

Mplayer依赖于以下程序:

- zlib: 包含了常用的压缩程序的动态链接库
- libmad: 高品质全定点算法的 MPEG 音频解码库
- ffmpeg: 久负盛名的开源视频编解码程序

其中，Mplayer在配置时会自动下载需要的ffmpeg源码，因此不需要提前准备ffmpeg

关于OSS与ALSA

OSS在绝大多数的设备中已经弃用，即使需要使用/dev/dsp设备文件进行操作，也可以直接在配置内核时选择使用ALSA而不是OSS，因为ALSA会提供一个/dev/dsp设备文件以支持仅能使用OSS的旧软件。

关于源码的交叉编译与部署

从源码安装程序大部分需要经过以下流程:

1. 运行源码提供的配置脚本，对Makefile进行配置。大部分的软件源码会将这个脚本命名为`configure`，脚本可以用于指定编译器、库文件链接方式、目标安装位置、启用/禁用功能等多种用途，在配置脚本运行结束后，会生成可用的Makefile。
2. 使用`make`命令调用Makefile进行程序、库文件等的编译。
3. 使用`make install`命令调用Makefile将程序运行时所需要的二进制文件、库文件、头文件等复制到目标位置，并进行一些其他操作如`stripe`、写入环境变量等。

但是整个程序编译部署能否成功极度依赖于配置脚本的正确与否，而这个脚本往往内容复杂，很难即使发现错误并进行纠正，尤其是在交叉编译的情况下，部分软件(如MPlayer)的配置脚本只考虑了编译器的交叉编译，而对于`stripe`和`ld`等命令完全没有更换成交叉编译的版本，从而无法完成安装。实际上，在下面每一个软件的移植中，都遇到了配置脚本的问题带来的挑战。

不过对于大多数软件，`make install`步骤只是用于整理压缩操作，因此如果在这一步失败，可以直接使用编译后的目录中的二进制文件运行，一般可以正常运行。

关于帧缓冲设备的兼容

之前发现，实验使用的开发板并不能很好地兼容运行时的分辨率修改问题，而MPlayer似乎会针对视频的完整分辨率尝试修改Frame Buffer的分辨率，由于之前的Frame Buffer问题，可能会导致分辨率实际上没有发生变化，但是此时再按照修改后的分辨率进行输出，就会发生错误，具体现象和暂时的解决方案在下文详细叙述。

实验流程

1. 准备工作

1.1 重新配置编译内核

使用`make menuconfig ARCH=arm`对内核进行自定义，需要保证如下条件满足：

1. Sound card support -> Advanced Linux Sound Architecture处于启用状态，这样系统运行的时候就可以通过调用ALSA API或者OSS来播放声音。
2. USB sound devices -> USB Audio/MIDI driver处于启用状态，这样系统可以使用USB声卡进行播放。

配置完成后重新编译内核。

至此，所有准备工作完成，使用新内核和根文件系统启动。

2. 移植zlib

1. 使用`apt-src`命令拉取源码：

```
$ apt-src install zlib
```

2. 进入源码文件夹，进行配置：

```
$ CC=arm-linux-gnueabihf-gcc ./configure --prefix=/home/minaduki/Desktop/Bea
```

其中`CC`用于指定当前环境下的编译器，`--prefix`字段用于配置`make install`安装的路径。

如果不配置`--prefix`字段，则默认会安装到`/usr/shared`目录下，这样就会造成系统原有的程序不能工作。

3. 使用GNU/make编译并安装程序：

```
$ make  
$ make install
```

值得一提的是，虽然我一开始在上面指定的路径中是`zlib`，但有的时候安装的路径会和预设的不一样，推测应该是配置脚本中存在一些bug的缘故。

查看安装路径，可以看到三个目录，分别存放头文件和库文件，以及一些share文件：

```
total 0  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 .  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 ..  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 include
```

```
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 lib  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 share
```

至此， zlib准备就绪。

3. 移植libmad

1. 使用apt-src命令拉取源码：

```
$ apt-src install libmad
```

2. 进入源码文件夹，进行配置：

```
$ ./configure --enable-fpm=arm --host=arm-linux-gnueabihf --enable-speed --p
```

其中CC用于指定当前环境下的编译器，--prefix字段用于配置make install安装的路径，--host字段用于指出交叉编译工具链的前缀，--enable-fpm=arm则用于启用硬件浮点运算并指定硬件浮点运算单元的平台。

3. 使用GNU/make编译并安装程序：

在开始编译前，应该先打开Makefile文件，寻找所有的-fforce-mem字段并删除，在新版本的arm-linux-gnueabihf-gcc中，这个字段已经不再受支持，删去它不会影响编译效果。

```
$ make  
$ make install
```

查看安装路径，可以看到两个目录，分别存放头文件和库文件：

```
total 0  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 .  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 ..  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 include  
drwxrwxrwx 1 minaduki minaduki 512 Jan 12 22:07 lib
```

至此， libmad准备就绪。

4. 移植MPlayer

1. 使用apt-src命令拉取源码，或者直接前往[官方网站](#)下载：

```
$ apt-src install mplayer
```

2. 进入源码文件夹，进行配置：

```
$ ./configure --cc=arm-linux-gnueabihf-gcc --target=arm-linux-gnueabihf --er
```

其中--cc字段用于指定当前环境下的编译器，--prefix字段用于配置make install安装的路径，--target字段用于指出交叉编译工具链的前缀，--extra-cflags和--extra-ldflags分别用于指定之前编译的 zlib 和 libmad 软件的头文件和库文件地址。其他的字段基本上都与功能和平台支持有关，不再赘述。

3. 使用GNU/make编译并安装程序：

```
$ make  
$ make install
```

在执行make install时，会发生报错，提示stripe程序遇到了无法识别的符号。这是因为配置脚本在配置Makefile时，没有去调用arm-linux-gnueabihf对应的stripe，而是调用了上位机的stripe，自然无法继续。

由于Makefile文件规模过大，而且用搜索功能搜索Makefile找不到stripe有关的字段，因此放弃install，直接使用编译完成的源码目录中的二进制文件运行：

```
$ ls -al mplayer  
-rwxrwxrwx 1 minaduki minaduki 18303436 Jan 13 23:33 mplayer  
$ file mplayer  
mplayer: ELF 32-bit LSB executable, ARM, EABI5 version 1 (GNU/Linux), static
```

在配置脚本中有一个字段是--enable-static，指使用静态库链接，不是必须选项，由于这里没有办法进行install，因此如果使用动态链接的话，要一个个找到所有的动态链接库拷贝到单片机上，容易产生遗漏，因此索性使用静态链接。

但是同时，这样生成的二进制文件大小高达18MiB，虽然我使用的是NFS文件系统，相对不受大小限制，但是通常在嵌入式系统中存储空间不带，因此重复利用是很重要的，所以虽然理论来说静态链接的mplayer不需要库文件也可以运行，但是为了展示移植软件的完整流程，下文依旧会进行库文件的部署。

至此，Mplayer(算是)准备就绪。

4. 部署程序

将安装路径下两个软件的lib目录下的所有文件拷贝到根文件系统的/lib目录下(如果不考虑二次开发可以只复制动态链接库，事实上都用上交叉编译了谁会想不开在开发板上二次开发，二次开发多半还是在上位机玩交叉编译)。

将安装路径下两个软件的include目录下的所有文件拷贝到根文件系统的/usr/include目录下，用于支持在开发板上进行二次开发。

将编译完成后的整个Mplayer源码目录拷贝至NFS共享目录。

5. 运行程序

查看使用方法

切换到Mplayer所在的目录，直接运行mplayer程序，检查一下程序能否正常运行并了解程序用法，程序输出如下：

```
/mnt/MPlayer-1.4 # ./mplayer
MPlayer 1.4-9 (C) 2000-2019 MPlayer Team
Usage: mplayer [options] [url|path/]filename

Basic options: (complete list in the man page)
-vo <drv>      select video output driver ('-vo help' for a list)
-ao <drv>      select audio output driver ('-ao help' for a list)
vcd://<trackno> play (S)VCD (Super Video CD) track (raw device, no mount)
-alang/-slang   select DVD audio/subtitle language (by 2-char country code)
-ss <position> seek to given (seconds or hh:mm:ss) position
-nosound        do not play sound
-fs             fullscreen playback (or -vm, -zoom, details in the man page)
-x <x> -y <y> set display resolution (for use with -vm or -zoom)
-sub <file>     specify subtitle file to use (also see -subfps, -subdelay)
-playlist <file> specify playlist file
-vid x -aid y select video (x) and audio (y) stream to play
-fps x -srate y change video (x fps) and audio (y Hz) rate
-pp <quality>  enable postprocessing filter (details in the man page)
-framedrop     enable frame dropping (for slow machines)

Basic keys: (complete list in the man page, also check input.conf)
<- or ->       seek backward/forward 10 seconds
down or up      seek backward/forward 1 minute
pgdown or pgup  seek backward/forward 10 minutes
< or >         step backward/forward in playlist
p or SPACE     pause movie (press any key to continue)
q or ESC        stop playing and quit program
+ or -          adjust audio delay by +/- 0.1 second
o               cycle OSD mode: none / seekbar / seekbar + timer
* or /          increase or decrease PCM volume
x or z          adjust subtitle delay by +/- 0.1 second
r or t          adjust subtitle position up/down, also see -vf expand

* * * SEE THE MAN PAGE FOR DETAILS, FURTHER (ADVANCED) OPTIONS AND KEYS * *
```

可以认为Mplayer安装配置成功，可以正常运行。

播放音频文件

使用mplayer播放一个普通的波形声音文件，连接耳机到USB声卡上，欣赏效果。

```
/mnt/MPlayer-1.4 # ./mplayer ../Guardians.wav
```

声音文件属性如下：

```
$ file Guardians.wav
Guardians.wav: RIFF (little-endian) data, WAVE audio, Microsoft PCM, 16 bit,
```

播放效果很好，音频输出正常，没有杂音或者缺失，播放流畅没有卡顿，终端输出内容如下：

```
MPlayer 1.4-9 (C) 2000-2019 MPlayer Team

Playing ../Guardians.wav.
libavformat version 58.27.102 (internal)
Audio only file format detected.
Load subtitles in ../
=====
Opening audio decoder: [pcm] Uncompressed PCM audio decoder
AUDIO: 48000 Hz, 2 ch, s16le, 1536.0 kbit/100.00% (ratio: 192000->192000)
Selected audio codec: [pcm] afm: pcm (Uncompressed PCM)
=====
A0: [oss] 48000Hz 2ch s16le (2 bytes per sample)
Video: no video
Starting playback...
A: 8.6 (08.5) of 75.0 (01:15.0) 1.1%
```



可以通过数字键9和0调整音量。

播放本地视频文件

使用mplayer播放一个大小较小的视频文件，连接耳机到USB声卡上播放声音，连接HDMI显示器显示画面。

在播放前，要使用fbset将Frame Buffer的分辨率设定为一个值，防止Mplayer设定失败而无法播放。

```
/mnt/MPlayer-1.4 # fbset -g 640 480 640 480 16
/mnt/MPlayer-1.4 # ./mplayer ../dog.mp4
```



终端输出如下：

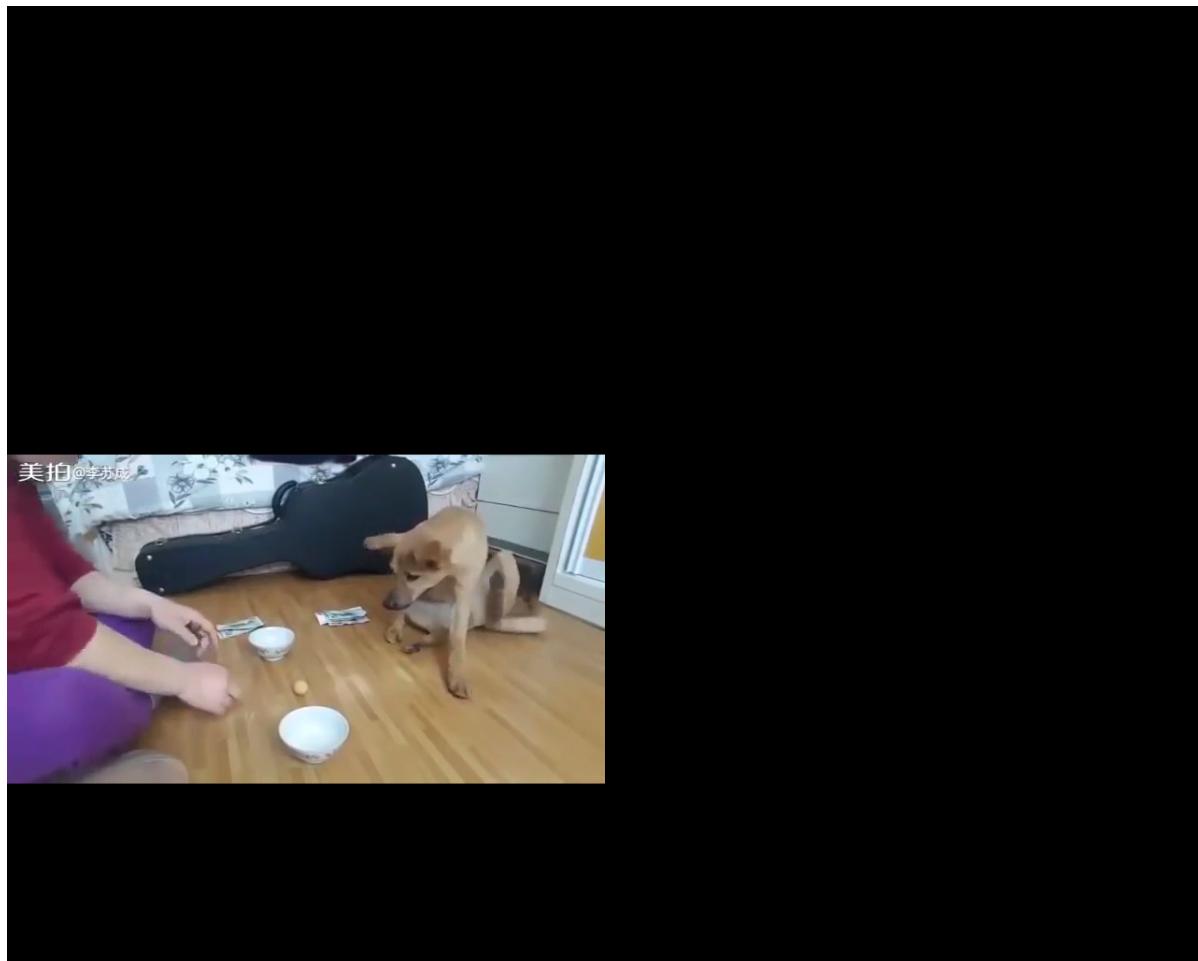
```
MPlayer 1.4-9 (C) 2000-2019 MPlayer Team

Playing ../dog.mp4.
libavformat version 58.27.102 (internal)
libavformat file format detected.
[mov,mp4,m4a,3gp,3g2,mj2 @ 0xf3f278]Protocol name not provided, cannot determine
if input is local or a network protocol, buffers and access patterns cannot
be configured optimally without knowing the protocol
[ 192.943290] tilcdc 4830e000.lcdc: tilcdc_crtc_irq(0x00000020): FIFO under
[ 192.950490] tilcdc 4830e000.lcdc: tilcdc_crtc_irq(0x00000104): Sync lost
[lavf] stream 0: video (h264), -vid 0
[lavf] stream 1: audio (aac), -aid 0, -alang und
VIDEO: [H264] 640x352 24bpp 24.000 fps 329.3 kbps (40.2 kbyte/s)
=====
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
libavcodec version 58.51.100 (internal)
Selected video codec: [ffvh264] vfm: ffmpeg (FFmpeg H.264)
=====
```

```
Clip info:  
major_brand: mp42  
minor_version: 0  
compatible_brands: isommp42  
creation_time: 2018-10-08T12:59:25.000000Z  
Load subtitles in ../  
=====  
Opening audio decoder: [ffmpeg] FFmpeg/libavcodec audio decoders  
AUDIO: 44100 Hz, 2 ch, floatle, 96.0 kbit/3.40% (ratio: 11999->352800)  
Selected audio codec: [ffaac] afm: ffmpeg (FFmpeg AAC (MPEG-2/MPEG-4 Audio))  
=====  
AO: [oss] 44100Hz 2ch s16le (2 bytes per sample)  
Starting playback...  
Could not find matching colorspace - retrying with -vf scale...  
Opening video filter: [scale]  
Movie-Aspect is 1.82:1 - prescaling to correct movie aspect.  
[swscaler @ 0xf5e900]bicubic scaler, from yuv420p to rgb565le using C  
[swscaler @ 0xf5e900]No accelerated colorspace conversion found from yuv420p  
rgb565le.  
[swscaler @ 0xf5e900]using unscaled yuv420p -> rgb565le special converter  
VO: [fbdev] 640x352 => 640x352 BGR 16-bit  
A: 0.4 V: 0.2 A-V: 0.192 ct: 0.000 0/ 0 ??% ??% ??,% 4 0
```



可以播放视频，但是屏幕在不该出现视频的部分有闪屏现象。



经过后续步骤发现，mplayer的拉伸画面并不是将视频拉伸至填满画面，而是使用黑色填满视频外的画面部分。

播放网络视频文件(HTTP协议)

使用使用mplayer在线播放同一个视频文件，连接耳机到USB声卡上播放声音，连接HDMI显示器显示画面。

```
/mnt/MPlayer-1.4 # ./mplayer http://101.132.47.121:6888/dog.mp4
```

由于没有配置DNS相关信息，因此没有办法指定域名，必须手动输入IP地址，Mplayer会访问IP地址进行播放。

```
MPlayer 1.4-9 (C) 2000-2019 MPlayer Team

Playing http://101.132.47.121:6888/dog.mp4.
Resolving 101.132.47.121 for AF_INET6...

Couldn't resolve name for AF_INET6: 101.132.47.121
Connecting to server 101.132.47.121[101.132.47.121]: 6888...

Cache size set to 320 KBytes
Cache fill: 0.00% (0 bytes)

libavformat version 58.27.102 (internal)
libavformat file format detected.
[mov,mp4,m4a,3gp,3g2,mj2 @ 0xf3f278]Protocol name not provided, cannot determine
if input is local or a network protocol, buffers and access patterns cannot
be configured optimally without knowing the protocol
[lavf] stream 0: video (h264), -vid 0
[lavf] stream 1: audio (aac), -aid 0, -alang und
VIDEO: [H264] 640x352 24bpp 24.000 fps 329.3 kbps (40.2 kbyte/s)
=====
Opening video decoder: [ffmpeg] FFmpeg's libavcodec codec family
libavcodec version 58.51.100 (internal)
Selected video codec: [ffh264] vfm: ffmpeg (FFmpeg H.264)
=====
Clip info:
major_brand: mp42
minor_version: 0
compatible_brands: isommp42
creation_time: 2018-10-08T12:59:25.000000Z
=====
Opening audio decoder: [ffmpeg] FFmpeg/libavcodec audio decoders
AUDIO: 44100 Hz, 2 ch, floatle, 96.0 kbit/3.40% (ratio: 11999->352800)
Selected audio codec: [ffaac] afm: ffmpeg (FFmpeg AAC (MPEG-2/MPEG-4 Audio))
=====
AO: [oss] 44100Hz 2ch s16le (2 bytes per sample)
Starting playback...
Could not find matching colorspace - retrying with -vf scale...
Opening video filter: [scale]
Movie-Aspect is 1.82:1 - prescaling to correct movie aspect.
[swscaler @ 0xf5e900]bicubic scaler, from yuv420p to rgb565le using C
[swscaler @ 0xf5e900]No acc[ 856.869871] tilcdc 4830e000.lcdc: tilcdc_crtc_
0x00000020): FIFO underflow
elerated colorspace conversion found from yuv420p to rgb565le.
[ 856.884982] tilcdc 4830e000.lcdc: tilcdc_crtc_irq(0x00000004): Sync lost
```

```
[swscaler @ 0xf5e900] using unscaled yuv420p -> rgb565le special [ 856.89639
ilcdc 4830e000.lcdc: tilcdc_crtc_irq(0x00000004): Sync lost
converter
VO: [fbdev] 640x352 => 640x352 BGR 16-bit
A: 9.0 V: 9.0 A-V: 0.000 ct: 0.000 0/ 0 40% 16% 10.9% 15 0 45%
```

相比本地播放，没有肉眼可见的画质/帧率损失，播放效果可以接受。

小结

关于帧缓冲设备与性能差异

为了验证是否是由于Frame Buffer的配置问题导致的视频播放问题，使用一个Raspberry Pi Zero WH移植Mplayer，播放媒体进行测试。树莓派中系统版本为：

```
$ uname -a
Linux raspberrypi 4.19.75+ #1270 Tue Sep 24 18:38:54 BST 2019 armv6l GNU/Lin
```

zlib和libmad在树莓派系统中已经包含，Mplayer使用同样的源码交叉编译。

播放视频，音视频输出正常，没有花屏现象，同时树莓派可以通过HDMI输出音频，但是对Frame Buffer截屏输出发现，设备的Frame Buffer存储了三帧的视频，在截取后没法转换成图片，但是的确可以正常播放：

```
frame= 3 fps=0.0 q=-0.0 Lsize=N/A time=00:00:00.08 bitrate=N/A speed=1.47
video:391kB audio:0kB subtitle:0kB other streams:0kB global headers:0kB muxi
Conversion failed!
```

因此推测可能的确是由于显示驱动的问题，导致在BeagleBone上视频播放效果不尽人意。

另外，在树莓派上可以播放对资源要求更大的视频而不会卡顿，虽然会提示系统资源占用过高，但是播放效果很好：

```
*****
*** Your system is too SLOW to play this! ***
*****
```

```
Possible reasons, problems, workarounds:
- Most common: broken/buggy _audio_ driver
  - Try -ao sdl or use the OSS emulation of ALSA.
  - Experiment with different values for -autosync, 30 is a good start.
- Slow video output
  - Try a different -vo driver (-vo help for a list) or try -framedrop!
- Slow CPU
  - Don't try to play a big DVD/DivX on a slow CPU! Try some of the lavdopts
    e.g. -vfm ffmpeg -lavdopts lowres=1:fast:skiploopfilter=all.
- Broken file
  - Try various combinations of -nobps -ni -forceidx -mc 0.
- Slow media (NFS/SMB mounts, DVD, VCD etc)
```

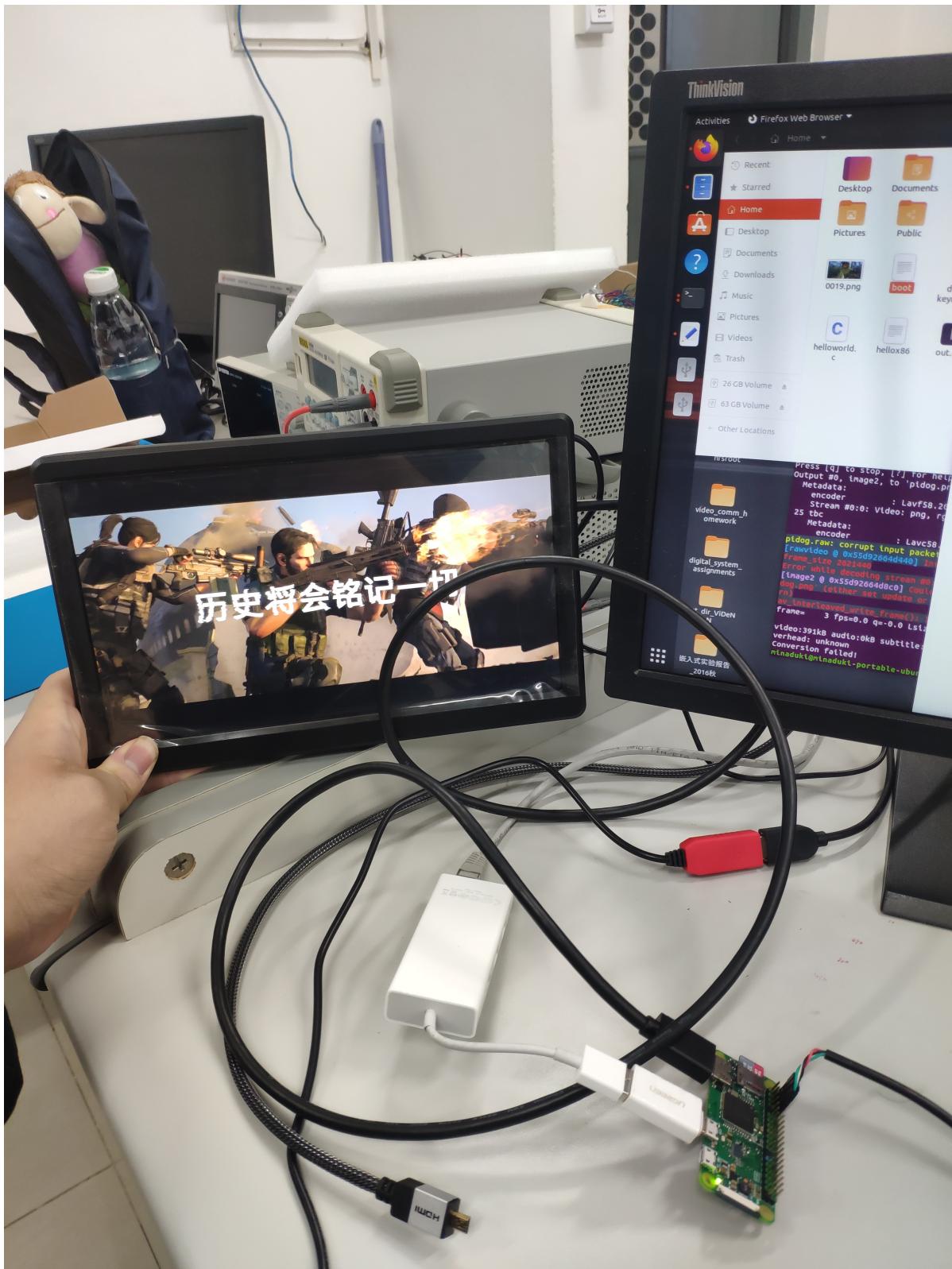
```
- Try -cache 8192.  
- Are you using -cache to play a non-interleaved AVI file?  
- Try -nocache.  
Read DOCS/HTML/en/video.html for tuning/speedup tips.  
If none of this helps you, read DOCS/HTML/en/bugreports.html.
```



但是同时，我发现，使用Mplayer播放器播放更大的视频会卡顿，但是使用VLC就很流畅：
(由于无法直接截屏，这里原本应有的图片没法放上来了)

```
pi@raspberrypi:~$ cvlc http://www.minaduki.cn:6888/td2.flv  
VLC media player 3.0.8 Vetinari (revision 3.0.8-0-gf350b6b5a7)
```





我一开始以为是我配置的问题，但是后来使用apt直接安装Mplayer(版本1.3)，也依然卡顿。由于两者用的编解码库都是ffmpeg，因此我猜想可能是因为两者调用了不同的显示渲染方式，从而产生了性能的差异。也许移植VLC能解决部分问题(如果没有被更复杂的依赖解决的话)。

还有一个很大的收获就是了解了包管理系统的重要性，管理依赖真的是个很复杂又很重要的任务。