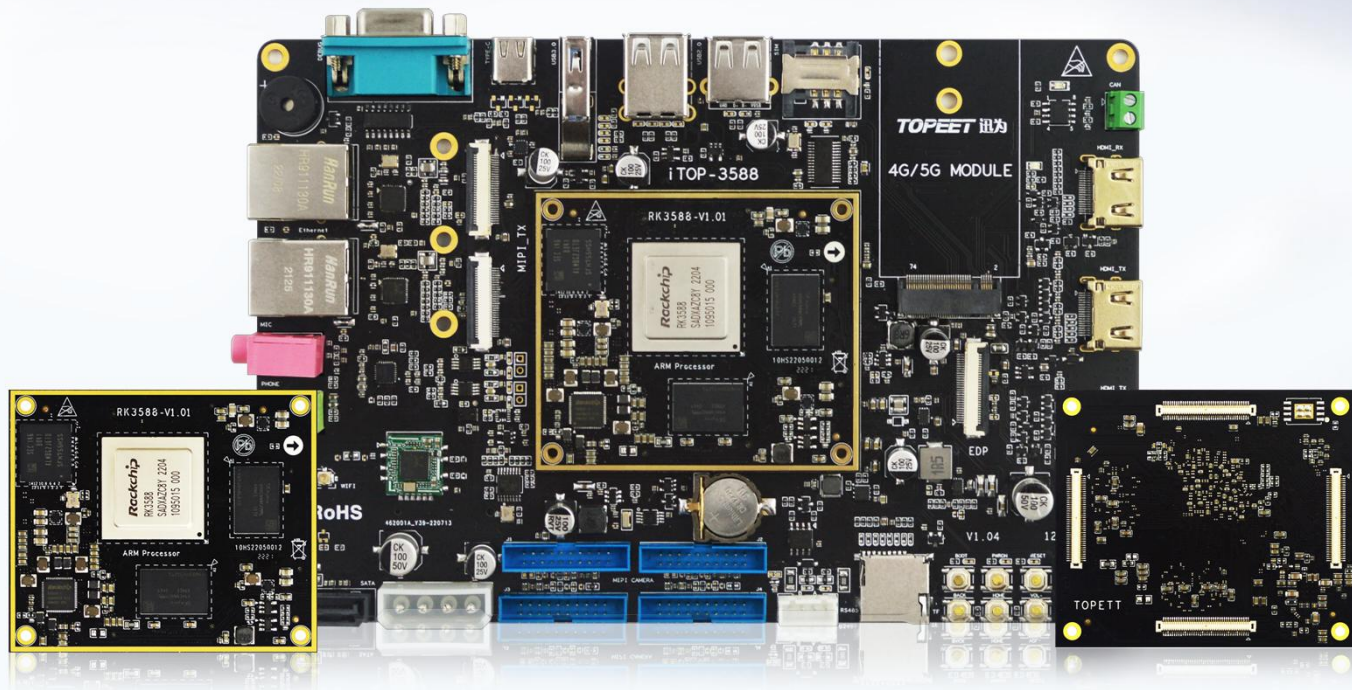


强大的 AI 能力 更快更强

超长供货周期 | 7X24 小时稳定运行 | 8K 视频编解码



iTOP-RK3588 开发板使用手册

八核 64 位 CPU | 主频 2.4GHz | NPU 算力 6T | 4800 安防级别 ISP



目录

更新记录	1
目录	2
版权声明	3
更多帮助	4
第 1 章 RTMP 推流之视频监控	5
1.1 搭建 RTMP 媒体服务器	5
1.1.1 Debian/ubuntu 系统	5
1.1.2 Buildroot 系统	9
1.2 使用 FFMpeg 进行推流	14
1.3 使用 VLCplayer 进行拉流	16
第二章 基于 Live555 的 RTSP 视频流服务器实现	18
2.1 Live555 源码简介	18
2.2 RTSP 推流	19

版权声明

本文档版权归北京迅为电子有限公司所有。未经本公司书面许可，任何单位和个人无权以任何形式复制、传播、转载本文档的任何内容，违者将被追究法律责任。

更多帮助

注意事项与维护

- ❖ 请注意和遵循标注在产品上的所有警示和指引信息；
- ❖ 请勿带电插拔核心板及外围模块；
- ❖ 使用产品之前，请仔细阅读本手册，并妥善保管，以备将来参考；
- ❖ 请使用配套电源适配器，以保证电压、电流的稳定；
- ❖ 请勿在冷热交替环境中使用本产品，避免结露损坏元器件；
- ❖ 请保持产品干燥，如果不慎被任何液体泼溅或浸润，请立刻断电并充分晾干；
- ❖ 请勿使用有机溶剂或腐蚀性液体清洗本产品；
- ❖ 请勿在多尘、脏乱的环境中使用本产品，如果长期不使用，请包装好本产品；
- ❖ 如果在震动场景使用，请做好核心板与底板的固定，避免核心板跌落损坏；
- ❖ 请勿在通电情况下，插拔核心板及外围模块(特别是串口模块)；
- ❖ 请勿自行维修、拆解本产品，如产品出现故障应及时联系本公司进行维修；
- ❖ 请勿自行修改或使用未经授权的配件，由此造成的损坏将不予保修；

资料的更新

为了确保您的资料是最新状态，请密切关注我们的动态，我们将会通过微信公众号和 QQ 群推送。

关注“迅为电子”微信公众号，不定期分享教程、资料和行业干货及产品一线资料。

迅为新媒体账号

官网：<https://www.topeetboard.com>

知乎：<https://www.zhihu.com/people/topeetabc123>

CSDN：<https://blog.csdn.net/BeijingXunWei>



售后服务政策

1. 如产品使用过程中出现硬件故障可根据售后服务政策进行维修
2. 服务政策：参见官方网售后服务说明
<https://www.topeetboard.com/sydymfl/Service/bx.html>

送修地址：

1. 地址：北京市海淀区永翔北路 9 号中国航发大厦三层
2. 联系人：迅为开发板售后服务部
3. 电话：010-85270716
4. 邮编：100094
5. 邮寄须知：建议使用顺丰、圆通或韵达，且不接受任何到付

技术支持范围

1. 了解产品的软、硬件资源提供情况咨询
2. 产品的软、硬件手册使用过程中遇到的问题
3. 下载和烧写更新系统过程中遇到的问题
4. 产品用户的资料丢失、更新后重新获取
5. 产品的故障判断及售后维修服务。

PS：（由于嵌入式系统知识范围广泛，我们无法保证对各种问题都能一一解答，部分内容无法供技术支持，只能提供建议。）

技术支持

1. 周一至周五：（法定节假日除外）
上午 9:00 ~ 11:30 / 下午 13:30 ~ 17:30
2. QQ 技术交流群：
824412014
822183461
95631883
861311530

第 1 章 RTMP 推流之视频监控

RTMP 全称是 Real Time Messaging Protocol（实时消息传输协议），RTMP 是一种通讯协议。该协议基于 TCP，是一个协议族，包括 RTMP 基本协议及 RTMPT/RTMPS/RTMPE 等多种变种。RTMP 是一种设计用来进行实时数据通信的网络协议，主要用来在 Flash/AIR 平台和支持 RTMP 协议的流媒体/交互服务器之间进行音视频和数据通信。现在更流行于直播平台服务器的推流处理！

RTMP 的工作原理是通过建立和维护 RTMP 客户端和 RTMP 服务端之间的通信路径来实现快速、可靠的数据传输。为了维持连续传递并避免单次传输数据量过大，RTMP 采用了传输层封包和数据流切片的实现形式。消息被分成称为 Chunk 的最小传输单位，并按照 Chunk Size 进行划分和复用，从而实现对当前带宽的合理利用。有效的消息如果超出当前 Chunk Size，将会被拆分成多个分块分批传输。通过指定首个 Chunk 和后续 Chunk 类型，以及 Chunk Header 的其他标志性数据，接收端能够有效还原和执行被切割的消息。

1.1 搭建 RTMP 流媒体服务器

nginx-rtmp 是一个基于 nginx 的 RTMP 服务模块，是一个功能强大的流媒体服务器模块，它提供了丰富的功能和灵活的配置选项，适用于构建各种规模的流媒体平台和应用。无论是搭建实时视频直播平台、点播系统或多屏互动应用，Nginx-RTMP 都是一个值得考虑的选择。这里我们使用 Nginx 来搭建流媒体服务器。

1.1.1 Debian/ubuntu 系统

1. 安装 nginx+rtmp 运行所要用到的库和依赖环境

```
apt-get update
```

```
apt-get install build-essential libpcre3 libpcre3-dev libssl-dev zlib1g-dev openssl
```

2. 下载 nginx-1.20.2 源码（下载地址：<http://nginx.org/download/nginx-1.20.2.tar.gz>）和 nginx-rtmp-module（下载地址：<https://github.com/arut/nginx-rtmp-module>），网盘“iTOP-3588 开发板\02_【iTOP-RK3588 开发板】开发资料\09_Linux 系统开发配套资料\05_开发板推流配套资料”目录下提供了下载好的。

3. 将 nginx-1.20.2.tar.gz 和 nginx-rtmp-module-master.zip 拷贝到开发板，使用以下命令解压，如下图所示：

```
tar -vxf nginx-1.20.2.tar.gz
```

```
unzip nginx-rtmp-module-master.zip
```



```
root@topeet:/$ ls
bin    home    nginx-1.20.2    opt    sdcard    udisk
boot   lib      nginx-1.20.2.tar.gz  proc   snap      userdata
data   lost+found nginx-rtmp-module-master  root   srv        usr
dev    media    nginx-rtmp-module-master.zip  run    sys        var
etc    mnt      dem              sbin   tmp
root@topeet:/$
```

4. 进入 nginx 源码文件夹，使用以下命令进行手动编译安装。如下图所示：

```
cd nginx-1.20.2
./configure --prefix=/usr/local/nginx-1.20.2 --add-module=../nginx-rtmp-module-master
--with-http_ssl_module
make
make install
```

```
root@iTOP-RK3568:~/nginx-1.20.2 $
root@iTOP-RK3568:~/nginx-1.20.2 $ ./configure --prefix=/usr/local/nginx-1.20.2 --add-module=../nginx-rtmp-module-master --with-http_ssl_module
checking for OS
+ Linux 4.19.232 aarch64
checking for C compiler ... found
+ using GNU C compiler
+ gcc version: 8.3.0 (Debian 8.3.0-6)
checking for gcc -pipe switch ... found
checking for -Wl,-E switch ... found
checking for gcc builtin atomic operations ... found
checking for C99 variadic macros ... found
checking for gcc variadic macros ... found
checking for gcc builtin 64 bit byteswap ... found
checking for unistd.h ... found
checking for inttypes.h ... found
checking for limits.h ... found
checking for sys/filio.h ... not found
checking for sys/param.h ... found
checking for sys/mount.h ... found
checking for sys/statvfs.h ... found
checking for crypt.h ... found
checking for Linux specific features
checking for epoll ... found
checking for EPOLLRDHUP ... found
checking for EPOLLEXCLUSIVE ... found
checking for eventfd() ... found
checking for O_PATH ... found
checking for sendfile() ... found
checking for sendfile64() ... found
checking for sys/prctl.h ... found
checking for prctl(PR_SET_DUMPABLE) ... found
```

```
root@iTOP-RK3568:~/nginx-1.20.2 $
root@iTOP-RK3568:~/nginx-1.20.2 $ make
make -f objs/Makefile
make[1]: 进入目录"/root/nginx-1.20.2"
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx.o \
src/core/nginx.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_log.o \
src/core/nginx_log.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_palloc.o \
src/core/nginx_palloc.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_array.o \
src/core/nginx_array.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_list.o \
src/core/nginx_list.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_hash.o \
src/core/nginx_hash.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_buf.o \
src/core/nginx_buf.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_queue.o \
src/core/nginx_queue.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_output_chain.o \
src/core/nginx_output_chain.c
cc -c -pipe -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g -I src/core -I src/event -I src/event/modules -I src/os/unix -I ../nginx-rtmp-module-master -I objs \
-o objs/src/core/nginx_string.o \
src/core/nginx_string.c
```

```
root@iTOP-RK3568:~/nginx-1.20.2 $
root@iTOP-RK3568:~/nginx-1.20.2 $ make install
make -f objs/Makefile install
make[1]: 进入目录"/root/nginx-1.20.2"
test -d '/usr/local/nginx-1.20.2' || mkdir -p '/usr/local/nginx-1.20.2'
test -d '/usr/local/nginx-1.20.2/sbin' \
|| mkdir -p '/usr/local/nginx-1.20.2/sbin'
test ! -f '/usr/local/nginx-1.20.2/sbin/nginx' \
|| mv '/usr/local/nginx-1.20.2/sbin/nginx' \
   '/usr/local/nginx-1.20.2/sbin/nginx.old'
cp objs/nginx '/usr/local/nginx-1.20.2/sbin/nginx'
test -d '/usr/local/nginx-1.20.2/conf' \
|| mkdir -p '/usr/local/nginx-1.20.2/conf'
cp conf/koi-win '/usr/local/nginx-1.20.2/conf'
cp conf/koi-utf '/usr/local/nginx-1.20.2/conf'
cp conf/win-utf '/usr/local/nginx-1.20.2/conf'
test -f '/usr/local/nginx-1.20.2/conf/mime.types' \
|| cp conf/mime.types '/usr/local/nginx-1.20.2/conf'
cp conf/mime.types '/usr/local/nginx-1.20.2/conf/mime.types.default'
test -f '/usr/local/nginx-1.20.2/conf/fastcgi_params' \
|| cp conf/fastcgi_params '/usr/local/nginx-1.20.2/conf'
cp conf/fastcgi_params \
   '/usr/local/nginx-1.20.2/conf/fastcgi_params.default'
test -f '/usr/local/nginx-1.20.2/conf/fastcgi.conf' \
|| cp conf/fastcgi.conf '/usr/local/nginx-1.20.2/conf'
cp conf/fastcgi.conf '/usr/local/nginx-1.20.2/conf/fastcgi.conf.default'
test -f '/usr/local/nginx-1.20.2/conf/uwsgi_params' \
|| cp conf/uwsgi_params '/usr/local/nginx-1.20.2/conf'
cp conf/uwsgi_params \
   '/usr/local/nginx-1.20.2/conf/uwsgi_params.default'
test -f '/usr/local/nginx-1.20.2/conf/scgi_params' \
|| cp conf/scgi_params '/usr/local/nginx-1.20.2/conf'
cp conf/scgi_params \
   '/usr/local/nginx-1.20.2/conf/scgi_params.default'
test -f '/usr/local/nginx-1.20.2/conf/nginx.conf' \
|| cp conf/nginx.conf '/usr/local/nginx-1.20.2/conf/nginx.conf'
cp conf/nginx.conf '/usr/local/nginx-1.20.2/conf/nginx.conf.default'
test -d '/usr/local/nginx-1.20.2/logs' \
|| mkdir -p '/usr/local/nginx-1.20.2/logs'
test -d '/usr/local/nginx-1.20.2/logs' \
|| mkdir -p '/usr/local/nginx-1.20.2/logs'
test -d '/usr/local/nginx-1.20.2/html' \
|| cp -R html '/usr/local/nginx-1.20.2'
test -d '/usr/local/nginx-1.20.2/logs' \
|| mkdir -p '/usr/local/nginx-1.20.2/logs'
make[1]: 离开目录"/root/nginx-1.20.2"
root@iTOP-RK3568:~/nginx-1.20.2 $
```

5.新建文件夹，用于保存 nginx 所形成的 web 服务的文件，如下图所示：

```
mkdir /usr/share/nginx /usr/share/nginx/html /usr/share/nginx/html/live
touch /usr/share/nginx/html/index.html
chmod -R 0777 /usr/share/nginx/html/live
```

```
root@topeet:/nginx-1.20.2$
re/nginx/html/live-1.20.2$ mkdir /usr/share/nginx /usr/share/nginx/html /usr/sha
root@topeet:/nginx-1.20.2$ touch /usr/share/nginx/html/index.html
root@topeet:/nginx-1.20.2$ chmod -R 0777 /usr/share/nginx/html/live
```

6.配置 nginx 服务，主要是设置 web 服务的主目录和推流到的端口、临时文件夹。首先设定 web 服务的根目录和网页端口，修改/usr/local/nginx-1.20.2/conf/nginx.conf 文件，如下图所示：

```
vim /usr/local/nginx-1.20.2/conf/nginx.conf
```

```
root@topeet:/nginx-1.20.2$ vi /usr/local/nginx-1.20.2/conf/nginx.conf
#gzip on;

server {
    listen      80;
    server_name localhost;

    #charset koi8-r;

    #access_log logs/host.access.log main;

    location / {
        root    /usr/share/nginx/html;
        index   index.html index.htm;
    }

    #error_page 404            /404.html;

    # redirect server error pages to the static page /50x.html
    #
    error_page   500 502 503 504  /50x.html;
    location = /50x.html {
        root    html;
    }
}
```

7.然后添加以下内容设置 rtmp 服务器的监听端口，推流数据的存储位置等，如下图所示：

```
rtmp {
    server {
        listen 1096; #监听的端口（注意端口占用）
        chunk_size 4096;
        application live {
            live on;
            record off;
            allow play all;
        }
    }
}
```



```
root@topeet:/nginx-1.20.2$ vi /usr/local/nginx-1.20.2/conf/nginx.conf
#user root; 改为当前用户名
worker_processes 1;

#error_log logs/error.log;
#error_log logs/error.log notice;
#error_log logs/error.log info;

#pid logs/nginx.pid;

rtmp {
    server {
        listen 1096; #监听的端口
        chunk_size 4096;
        application live {
            live on;
            record off;
            allow play all;
        }
    }
}

events {
    worker_connections 1024;
}
```

8.启动 nginx，使用以下命令启动，启动成功如下图所示：

```
/usr/local/nginx-1.20.2/sbin/nginx -c /usr/local/nginx-1.20.2/conf/nginx.conf
ps -aux | grep nginx
```

```
root@topeet:/nginx-1.20.2$ /usr/local/nginx-1.20.2/sbin/nginx -c /usr/local/nginx-1.20.2/conf/nginx.conf
root@topeet:/nginx-1.20.2$ ps -aux | grep nginx
root    29022  0.0  0.0  15276  864 ?        Ss   03:02   0:00 nginx: master process /usr/local/nginx-1.20.2/sbin/nginx -c /usr/local/nginx-1.20.2/conf/nginx.conf
nobody  29023  0.0  0.0  16672  3592 ?        S    03:02   0:00 nginx: worker process
root    29025  0.0  0.0   6048   628 ttyFIQ0  S+   03:02   0:00 grep --color=auto nginx
```

1.1.2 Buildroot 系统

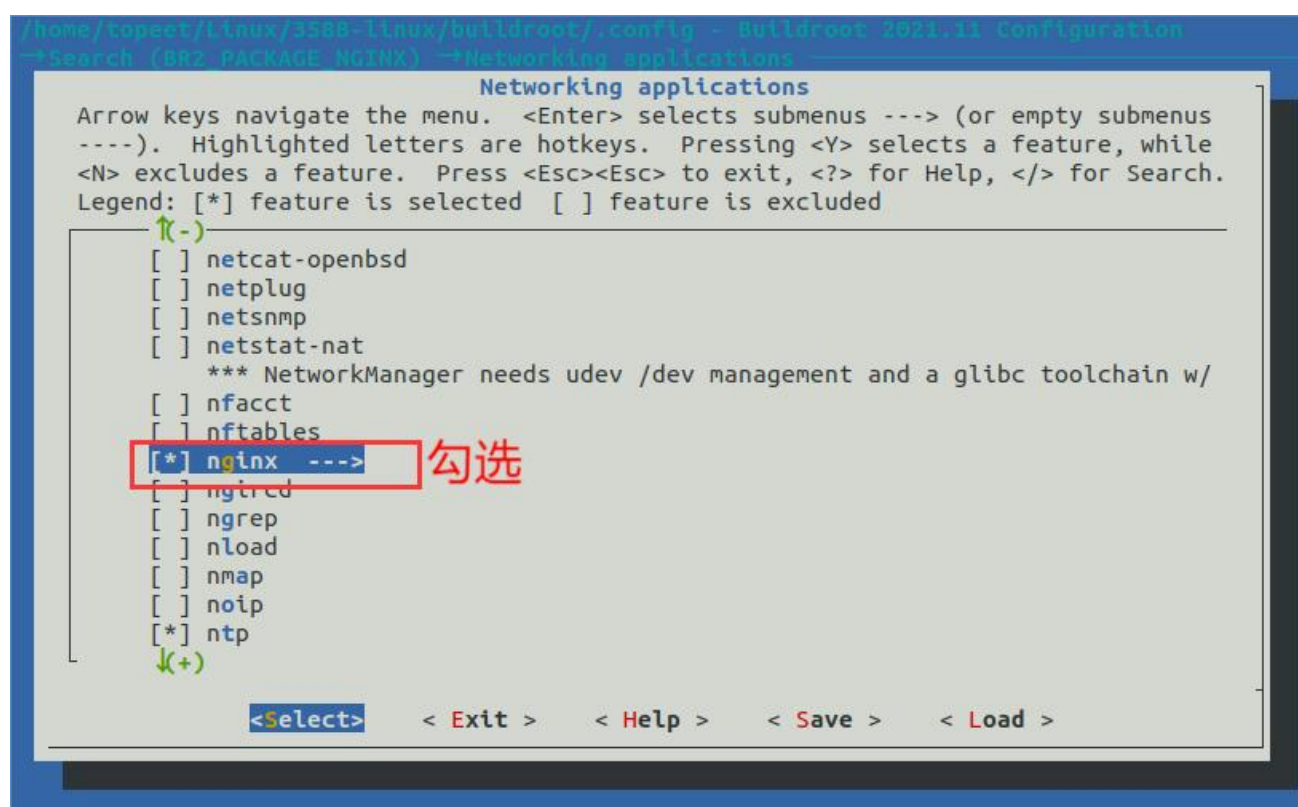
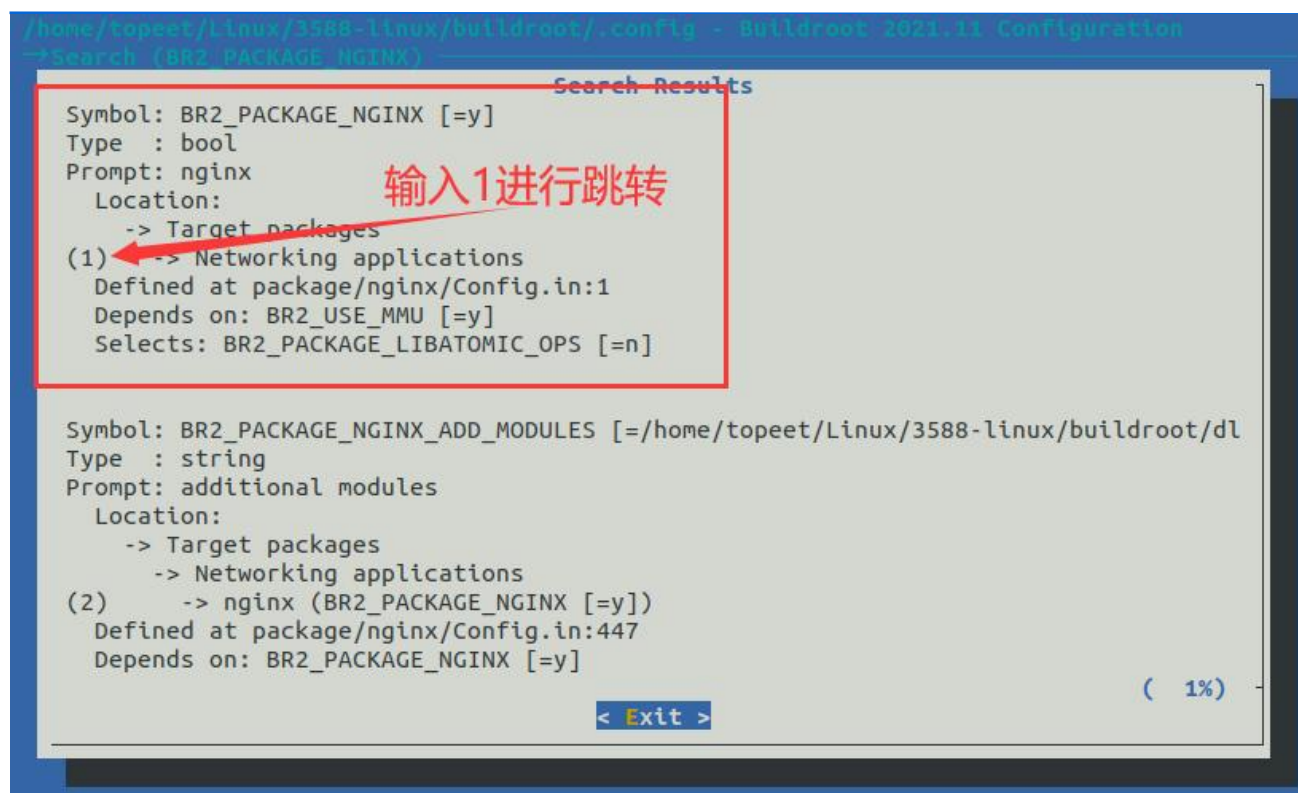
首先将网盘“iTOP-3588 开发板\02_【iTOP-RK3588 开发板】开发资料\09_Linux 系统开发配套资料\05_开发板推流配套资料”目录下的 nginx-rtmp-module-master.zip 拷贝到源码 buildroot/dl/nginx(没有则创建)目录下，并使用以下命令解压，并查看当前路径，如下图所示：

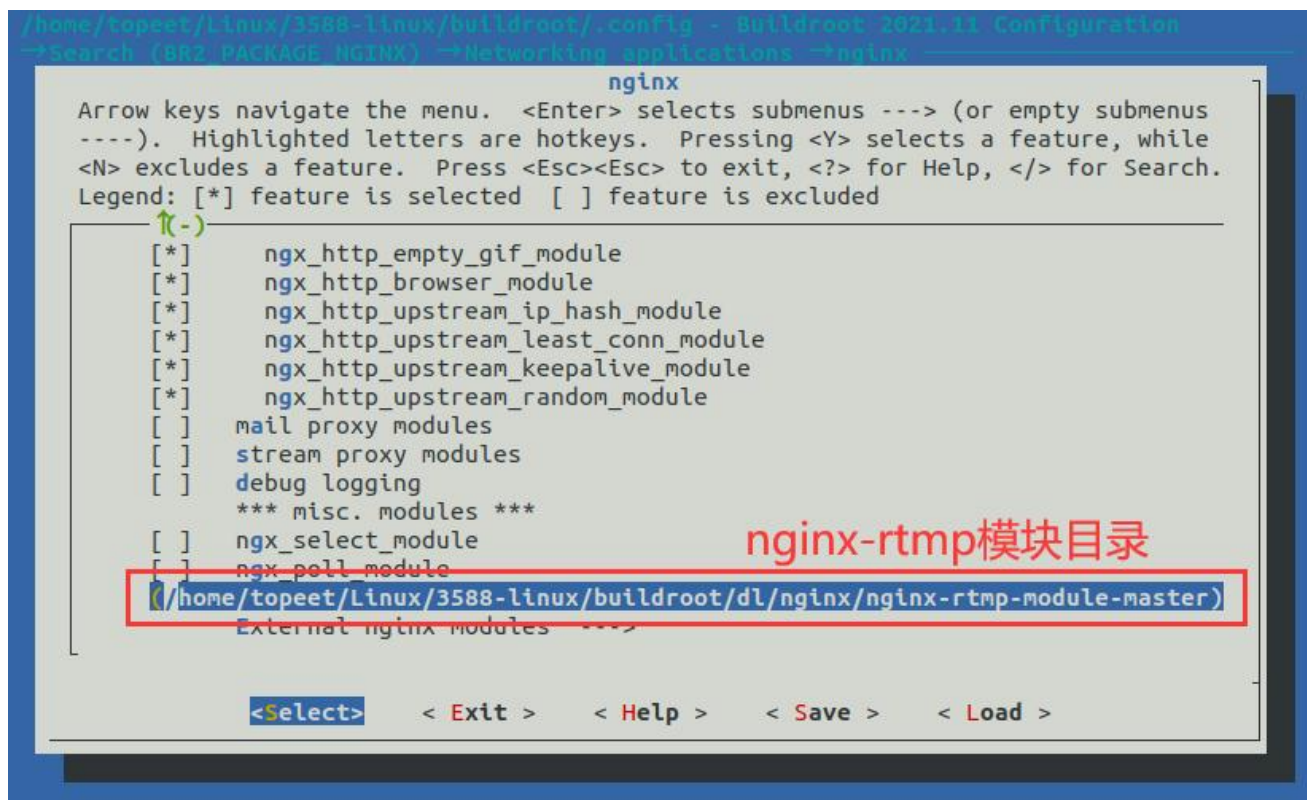
```
unzip nginx-rtmp-module-master.zip
cd nginx-rtmp-module-master
pwd
```

```
topeet@ubuntu:~/Linux/3588-linux/buildroot/dl/nginx$ cd nginx-rtmp-module-master/
topeet@ubuntu:~/Linux/3588-linux/buildroot/dl/nginx/nginx-rtmp-module-master$ pwd
/home/topeet/Linux/3588-linux/buildroot/dl/nginx/nginx-rtmp-module-master
topeet@ubuntu:~/Linux/3588-linux/buildroot/dl/nginx/nginx-rtmp-module-master$
```

Buildroot 系统默认没有勾选 Nginx，使用以下命令配置 buildroot，如下图所示：

```
cd buildroot
make rockchip_rk3588_defconfig
make menuconfig
```



配置完保存退出，使用以下命令保存修改至默认配置文件，然后重新编译 buildroot 烧写即可。

```
cp output/rockchip_rk3588/.config configs/rockchip_rk3588_defconfig
```

系统重新烧写启动后，使用以下命令查看，可以看到 Nginx 已经启动，如下图所示：

```
ps -aux | grep nginx
```

```

root@RK3588:/# ps -aux | grep nginx
root      977  0.0  0.0   3552   336 ?        Ss   08:07   0:00 nginx: master process /usr/sbin/nginx
www-data  978  0.0  0.0   3976   1588 ?        S    08:07   0:00 nginx: worker process
root     1532  0.0  0.0   3048   372 ttyFIQ0  S+   08:07   0:00 grep nginx
root@RK3588:/#
root@RK3588:/#
  
```

此时我们打开电脑浏览器，输入开发板的 IP 地址，如下图所示，表示我们的 nginx 已经正常工作了。

```

root@RK3588:/# ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 1A:6A:52:FD:32:18
          inet addr:192.168.1.212  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: 240e:341:d0a7:1500:9c0b:f94e:873b:6072/64 Scope:Global
          inet6 addr: fe80::9be5:2cf5:45b0:d77f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:335  errors:0  dropped:0  overruns:0  frame:0
          TX packets:119  errors:0  dropped:0  overruns:0  carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:34999 (34.1 KiB)  TX bytes:11570 (11.2 KiB)
          Interrupt:116

root@RK3588:/#
  
```



接下来我们使用以下命令打开 nginx 配置文件并添加以下内容，如下图所示：

```
vi /etc/nginx/nginx.conf
```

添加内容如下：

```
rtmp {  
    server {  
        listen 1096; #监听的端口（注意端口占用）  
        chunk_size 4096;  
        application live {  
            live on;  
            record off;  
            allow play all;  
        }  
    }  
}
```

```
#error_log logs/error.log;  
#error_log logs/error.log notice;  
#error_log logs/error.log info;  
#pid logs/nginx.pid;  
  
rtmp {  
    server {  
        listen 1096;  
        chunk_size 4096;  
        application live {  
            live on;  
            record off;  
            allow play all;  
        }  
    }  
}  
  
events {  
    worker_connections 1024;  
}
```


添加完成之后输入以下命令重启 nginx 即可，如下图所示：

```
/etc/init.d/S50nginx restart
```

```
root@RK3588:/#  
root@RK3588:/# /etc/init.d/S50nginx restart  
Stopping nginx...  
stopped /usr/sbin/nginx (pid 514)  
Starting nginx...  
root@RK3588:/#
```

1.2 使用 FFMpeg 进行推流

Debian/Ubuntu 系统使用以下命令安装 FFMpeg，如下图所示：

```
apt-get install ffmpeg
```

```
root@topeet:/$  
root@topeet:/$ apt-get install ffmpeg  
正在读取软件包列表... 完成  
正在分析软件包的依赖关系树  
正在读取状态信息... 完成  
ffmpeg 已经是最新版 (7:4.2.7-0ubuntu0.1)。  
升级了 0 个软件包，新安装了 0 个软件包，要卸载 0 个软件包，有 37 个软件包未被升级。  
root@topeet:/$  
root@topeet:/$
```

使用 ifconfig 查看开发板 ip 为 192.168.1.245 如下图所示：

```
root@topeet:/$ ifconfig eth1  
eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.168.1.245 netmask 255.255.255.0 broadcast 192.168.1.255  
    inet6 240e:341:d0a7:1500:3080:2309:2ed4:48c0 prefixlen 64 scopeid 0x0<global>  
    inet6 240e:341:d0a7:1500:d8c5:a8bb:489e:92b8 prefixlen 64 scopeid 0x0<global>  
    inet6 fe80::f212:7ca8:99a2:616a prefixlen 64 scopeid 0x20<link>  
    ether ae:a2:da:ac:65:d3 txqueuelen 1000 (Ethernet)  
    RX packets 47264 bytes 58739009 (58.7 MB)  
    RX errors 0 dropped 1051 overruns 0 frame 0  
    TX packets 4046 bytes 339739 (339.7 KB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
    device interrupt 115  
  
root@topeet:/$
```

使用 FFMpeg 推流一个 mp4 视频进行测试，作者将测试视频 test.mp4 放在了根目录下，可根据实际目录修改以下命令，如下图所示：

```
ffmpeg -re -stream_loop -1 -i /test.mp4 -f flv "rtmp://192.168.1.245:1096/live/test"
```

```
root@topeet:/$
245:1096/live/test'g -re -stream_loop -1 -i /test.mp4 -f flv "rtmp://192.168.1
ffmpeg version 4.2.7-0ubuntu0.1 Copyright (c) 2000-2022 the FFmpeg developers
built with gcc 9 (Ubuntu 9.4.0-1ubuntu1-20.04.1)
configuration: --prefix=/usr --extra-version=0ubuntu0.1 --toolchain=hardened --libdir=/usr/lib/aarch64-linux-gnu --incdir=/usr/include/aarch64-linux-gnu --arch=arm64
--enable-avresample --disable-filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-libaom --enable-libass --enable-libbluray --enable-libs2b --
--enable-libcodec2 --enable-libflite --enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm --enable-libjack --enable-libmp3la
libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-libsvg --enable-librubberband --enable-libshine --enable-libsnap --enable-libsoxr --enabl
able-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --enable-libwebp --enable-libx265 --enable-libxml2 --enabl
ble-libzvb1 --enable-lv2 --enable-omx --enable-opengl --enable-opencore --enable-opencore --enable-opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-libiec61883 --enable-chr
le-libx264 --enable-shared
libavutil 56. 31.100 / 56. 31.100
libavcodec 58. 54.100 / 58. 54.100
libavformat 58. 29.100 / 58. 29.100
libavdevice 58. 8.100 / 58. 8.100
libavfilter 7. 57.100 / 7. 57.100
libavresample 4. 0. 0 / 4. 0. 0
libswscale 5. 5.100 / 5. 5.100
libswresample 3. 5.100 / 3. 5.100
libpostproc 55. 5.100 / 55. 5.100
[mov,mp4,m4a,3gp,3g2,mj2 @ 0x5593a2d7a0] stream 0, timescale not set
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from '/test.mp4':
Metadata:
  major_brand : isom
  minor_version : 512
  compatible_brands : isomiso2avc1mp41
  artist : 房东的猫
  comment : 163 key(Don't modify):poXGw9c0rb4/fqFqJdENX49AMwLzUtK3R5YfIz+kc0mLVFXp/qibYUA1AZI8dooWB4dcWiDpy9MIoH4CT7qa44MRPU03hL7ZiDkiAwMtPnQeRxn3BKQxHhVkw+sY
faZKd7fLs8tms2wZrVCE0UaPZuacMWZ7Ga/F5BcA/GiFVA5ntLjp60XHEHm0473X6Scii5YHN+1wY
  title : 我是江小白
  encoder : Lavf56.40.101
Duration: 00:02:02.67, start: 0.000000, bitrate: 352 kb/s
Stream #0:0(eng): Video: h264 (Constrained Baseline) (avc1 / 0x31637661), yuv420p, 426x240 [SAR 1:1 DAR 71:40], 222 kb/s, 25 fps, 25 tbr, 12800 tbn, 50 tbc (default)
Metadata:
  handler_name : VideoHandler
Stream #0:1(eng): Audio: aac (LC) (mp4a / 0x6134706D), 44100 Hz, stereo, fltp, 120 kb/s (default)
Metadata:
  handler_name : SoundHandler
Stream #0:2: Video: mpeg (Baseline), yuvj420p(pc, bt470bg/unknown/unknown), 1920x1079 [SAR 72:72 DAR 1920:1079], 90k tbr, 90k tbn, 90k tbc (attached pic)
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> flv1 (flv))
  Stream #0:1 -> #0:1 (aac (native) -> mp3 (libmp3lame))
Press [q] to stop, [?] for help
Output #0, flv, to 'rtmp://192.168.1.245:1096/live/test':
Metadata:
  major_brand : isom
  minor_version : 512
  compatible_brands : isomiso2avc1mp41
  artist : 房东的猫
  comment : 163 key(Don't modify):poXGw9c0rb4/fqFqJdENX49AMwLzUtK3R5YfIz+kc0mLVFXp/qibYUA1AZI8dooWB4dcWiDpy9MIoH4CT7qa44MRPU03hL7ZiDkiAwMtPnQeRxn3BKQxHhVkw+sY
faZKd7fLs8tms2wZrVCE0UaPZuacMWZ7Ga/F5BcA/GiFVA5ntLjp60XHEHm0473X6Scii5YHN+1wY
  title : 我是江小白
  encoder : Lavf58.29.100
```

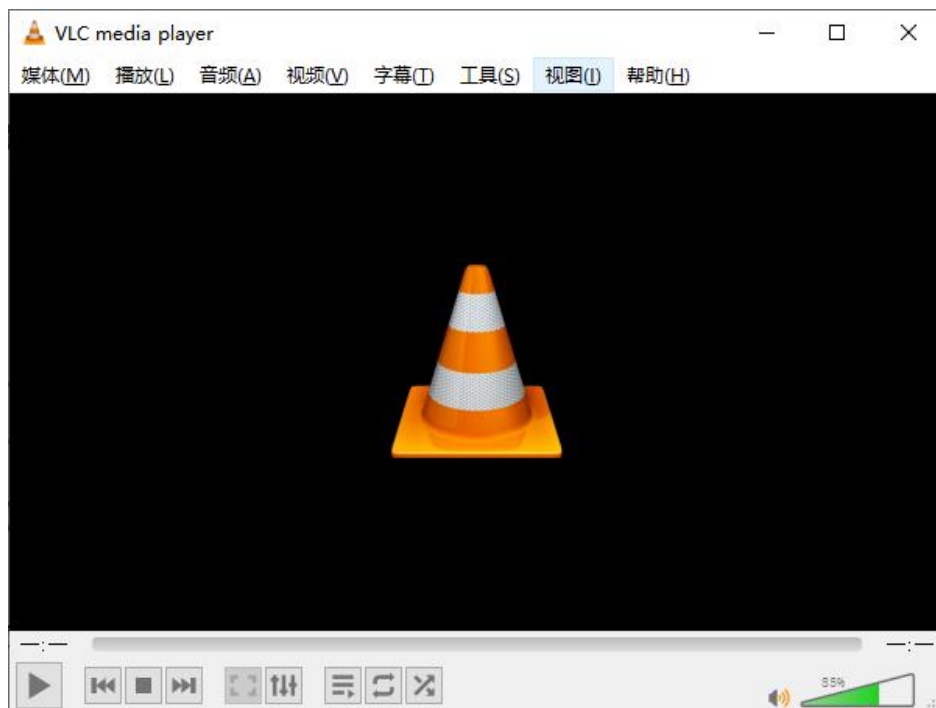
使用 FFmpeg 采集摄像头（测试 USB 摄像头没问题，可以使用 v4l2-ctl --list-devices 命令查看摄像头节点）视频数据将其发送给 nginx 流媒体服务器，执行以下命令：

```
ffmpeg -f v4l2 -video_size 640x480 -framerate 30 -i /dev/video21 -q 10 -f flv
rtmp://192.168.1.245:1096/live/test
```

```
root@topeet:/$
root@topeet:/$
root@topeet:/$
-q 10 -f flv rtmp://192.168.1.245:1096/live/test0 -framerate 30 -i /dev/video21
ffmpeg version 4.2.7-0ubuntu0.1 Copyright (c) 2000-2022 the FFmpeg developers
built with gcc 9 (Ubuntu 9.4.0-1ubuntu1-20.04.1)
configuration: --prefix=/usr --extra-version=0ubuntu0.1 --toolchain=hardened --libdir=/usr/lib/aarch64-linux-gnu --incdir=/usr/include/aarch64
--enable-avresample --disable-filter=resample --enable-avisynth --enable-gnutls --enable-ladspa --enable-libaom --enable-libass --enable-libblu
--enable-libcodec2 --enable-libflite --enable-libfontconfig --enable-libfreetype --enable-libfribidi --enable-libgme --enable-libgsm --enable-l
libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-libsvg --enable-librubberband --enable-libshine --enable-libsnap --enable-libs
able-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libvpx --enable-libwavpack --enable-libwebp --enable-libx265 --enable-
ble-libzvb1 --enable-lv2 --enable-omx --enable-opengl --enable-opencore --enable-opencore --enable-opengl --enable-sdl2 --enable-libdc1394 --enable-libdrm --enable-
le-libx264 --enable-shared
libavutil 56. 31.100 / 56. 31.100
libavcodec 58. 54.100 / 58. 54.100
libavformat 58. 29.100 / 58. 29.100
libavdevice 58. 8.100 / 58. 8.100
libavfilter 7. 57.100 / 7. 57.100
libavresample 4. 0. 0 / 4. 0. 0
libswscale 5. 5.100 / 5. 5.100
libswresample 3. 5.100 / 3. 5.100
libpostproc 55. 5.100 / 55. 5.100
Input #0, video4linux2,v4l2, from '/dev/video21':
Duration: N/A, start: 6293.792863, bitrate: 147456 kb/s
Stream #0:0: Video: rawvideo (YUY2 / 0x32595559), yuyv422, 640x480, 147456 kb/s, 30 fps, 30 tbr, 1000k tbn, 1000k tbc
Stream mapping:
  Stream #0:0 -> #0:0 (rawvideo (native) -> flv1 (flv))
Press [q] to stop, [?] for help
Output #0, flv, to 'rtmp://192.168.1.245:1096/live/test':
Metadata:
  encoder : Lavf58.29.100
Stream #0:0: Video: flv1 (flv) ([2][0][0][0] / 0x0002), yuv420p, 640x480, q=2-31, 200 kb/s, 30 fps, 1k tbn, 30 tbc
Metadata:
  encoder : Lavc58.54.100 flv
Side data:
  cnb: bitrate max/min/avg: 0/0/200000 buffer size: 0 vbv delay: 0
```

1.3 使用 VLCplayer 进行拉流

我们将 Windows 主机作为拉流端，使用 VLC 软件进行拉流，VLC 软件可以在网盘资料“iTOP-3588 开发板\02_【iTOP-RK3588 开发板】开发资料\09_Linux 系统开发配套资料\05_开发板推流配套资料”目录下找到，大家可以自行安装好。安装好之后打开 VLC，如下图所示：



点击左上角媒体->打开网络串流，弹出如下界面输入以下内容，如下图所示：



点击“播放”即可从 RTMP 流媒体服务器拉取视频数据进行播放，如下所示：


```
libavresample 4. 0. 0 / 4. 0. 0
libswscale 5. 5.100 / 5. 5.100
libswresample 3. 5.100 / 3. 5.100
libpostproc 55. 5.100 / 55. 5.100
[mov,mp4,m4a,3gp,3g2,mj2 @ 0x5593a2d7a0] stream 0, timescale not set
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from '/test.mp4':
Metadata:
  major_brand      : isom
  minor_version    : 512
  compatible_brands: isomiso2avc1mp41
  artist           : 房东的猫
  comment          : 163 key(Don't modify):poXGw9c0rb4/fqFqJdENX49AMwLzutK3R5Yfiz+kc0mLFVxp/qibyuA1AZI8dooW64dcWiDpy9MIoH4CT7qa44MRPU03hL7ZiDkiAwMtPnQeRxn3BKQxHhVvKw+sY+H5706n
  title            : 我是江小白
  encoder          : Lavf56.40.101
  Duration: 00:02:02.67, start: 0.000000, bitrate: 352 kb/s
  Stream #0:0(eng): Video: h264 (Constrained Baseline) (avc1 / 0x31637661), yuv420p, 426x240 [SAR 1:1 DAR 71:40], 222 kb/s, 25 fps, 25 tbr, 12800 tbn, 50 tbc (default)
  Metadata:
    handler_name    : VideoHandler
  Stream #0:1(eng): Audio: aac (LC) (mp4a / 0x61347060), 44100 Hz, stereo, fltp, 120 kb/s (default)
  Metadata:
    handler_name    : SoundHandler
  Stream #0:2: Video: mjpeg (Baseline), yuvj420p(pc, bt470bg/unknown/unknown), 704x576, 25 tbc (attached pic)
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> flv1 (flv))
  Stream #0:1 -> #0:1 (aac (native) -> mp3 (libmp3lame))
Press [q] to stop, [?] for help
Output #0, flv, to 'rtmp://192.168.1.245:1096/live/test':
Metadata:
  major_brand      : isom
  minor_version    : 512
  compatible_brands: isomiso2avc1mp41
  artist           : 房东的猫
  comment          : 163 key(Don't modify):poXGw9c0rb4/fqFqJdENX49AMwLzutK3R5Yfiz+kc0mLFVxp/qibyuA1AZI8dooW64dcWiDpy9MIoH4CT7qa44MRPU03hL7ZiDkiAwMtPnQeRxn3BKQxHhVvKw+sY+H5706n
  title            : 我是江小白
  encoder          : Lavf58.29.100
  Stream #0:0(eng): Video: flv1 (flv) ([2][0][0][0] / 0x0002), yuv420p, 426x240, 25 tbc (default)
  Metadata:
    handler_name    : VideoHandler
    encoder         : Lavc58.54.100 flv
  Side data:
    cpb: bitrate max/min/avg: 0/0/2000000 buffer size: 0 vbv_delay: -1
  Stream #0:1(eng): Audio: mp3 (libmp3lame) ([2][0][0][0] / 0x0002), 44100 Hz, stereo, fltp, 120 kb/s (default)
  Metadata:
    handler_name    : SoundHandler
    encoder         : Lavc58.54.100 libmp3lame
frame= 648 fps= 25 q=8.7 size= 1402kB time=00:00:26.12 bitrate= 439.5kbits/s speed=0.996x
```

```
root@topeet:/$
root@topeet:/$
root@topeet:/$
-g 10 -f flv rtmp://192.168.1.245:1096/live/test0 -framerate 30 -i /dev/video21
ffmpeg version 4.2.7-0ubuntu0.1 Copyright (c) 2000-2022 the FFmpeg developers
built with gcc 9 (Ubuntu 9.4.0-1ubuntu1-20.04.1)
configuration: --prefix=/usr --extra-version=0ubuntu0.1 --toolchain=hardened
--enable-avresample --disable-filter=resample --enable-avisynth --enable-gnutls
--enable-libcodec2 --enable-libflite --enable-libfontconfig --enable-libfreetype
libopenjpeg --enable-libopenmpt --enable-libopus --enable-libpulse --enable-librtmp
--enable-libtheora --enable-libtwolame --enable-libvidstab --enable-libvorbis --enable-libz
ble-libzstd --enable-lv2 --enable-omx --enable-opengl --enable-opencore-amrnb --enable-
le-libx264 --enable-shared
libavutil 56. 31.100 / 56. 31.100
libavcodec 58. 54.100 / 58. 54.100
libavformat 58. 29.100 / 58. 29.100
libavdevice 58. 8.100 / 58. 8.100
libavfilter 7. 57.100 / 7. 57.100
libavresample 4. 0. 0 / 4. 0. 0
libswscale 5. 5.100 / 5. 5.100
libswresample 3. 5.100 / 3. 5.100
libpostproc 55. 5.100 / 55. 5.100
Input #0, video4linux2,v4l2, from '/dev/video21':
Duration: N/A, start: 6293.792863, bitrate: 147456 kb/s
Stream #0:0: Video: rawvideo (YUY2 / 0x32595559), yuyv422, 640x480, 147456 kb/s, 25 fps, 25 tbr, 12800 tbn, 12800 tbc
Stream mapping:
  Stream #0:0 -> #0:0 (rawvideo (native) -> flv1 (flv))
Press [q] to stop, [?] for help
Output #0, flv, to 'rtmp://192.168.1.245:1096/live/test':
Metadata:
  encoder         : Lavf58.29.100
  Stream #0:0: Video: flv1 (flv) ([2][0][0][0] / 0x0002), yuv420p, 640x480, 25 tbc (default)
  Metadata:
    encoder         : Lavc58.54.100 flv
  Side data:
    cpb: bitrate max/min/avg: 0/0/2000000 buffer size: 0 vbv_delay: -1
[ 6309.837794] rk_gmac-dwmac felc0000.ethernet eth0: no phy at addr -164.8kbits/s speed= 1x
[ 6309.837841] rk_gmac-dwmac felc0000.ethernet eth0: stmmac_open: Cannot attach to PHY (error: -19)
[ 6309.852135] rk_gmac-dwmac felc0000.ethernet eth0: no phy at addr -1
[ 6309.852165] rk_gmac-dwmac felc0000.ethernet eth0: stmmac_open: Cannot attach to PHY (error: -19)
[ 6309.866914] rk_gmac-dwmac felc0000.ethernet eth0: no phy at addr -1
[ 6309.866944] rk_gmac-dwmac felc0000.ethernet eth0: stmmac_open: Cannot attach to PHY (error: -19)
[ 6309.876656] rk_gmac-dwmac felc0000.ethernet eth0: no phy at addr -1
[ 6309.876686] rk_gmac-dwmac felc0000.ethernet eth0: stmmac_open: Cannot attach to PHY (error: -19)
frame= 722 fps= 25 q=10.0 size= 1628kB time=00:00:28.83 bitrate= 460.4kbits/s speed= 1x
```

第二章 基于 Live555 的 RTSP 视频流服务器实现

live555 是一个 C++ 开发的流媒体项目，它主要由几个用于多媒体流的库组成，其官方网站地址为 <http://www.live555.com/>。live555 使用开放的标准协议 (RTP/RTCP, RTSP, SIP)，方便与其它标准的流媒体组件互操作。这些库可以为 Unix-like（包括 Linux 和 Mac OS X），Windows，和 QNX（及其它 POSIX 兼容系统）等系统进行编译，它们可以被用于构建流媒体应用。

live555 的库可以被用于处理 MPEG, H.265, H.264, H.263+, DV 或 JPEG 视频，及多种音频格式。它们还可以非常简单地扩展，以支持其它的音频或视频编解码格式，并可以被用于构建基本的 RTSP 或 SIP 客户端和服务端。

2.1 Live555 源码简介

live555 源码主要由八个部分组成: UsageEnvironment, BasicUsageEnvironment, groupsock, liveMedia, mediaServer, proxyServer, testProgs, WindowsAudioInputDevice。各个部分的简要说明如下。

1.UsageEnvironment 和 BasicUsageEnvironment

UsageEnvironment 中的 "UsageEnvironment" 和 "TaskScheduler" 类用于调度延迟的事件，为异步的读事件分配处理程序，以及输出错误/警告消息。UsageEnvironment 中的 "HashTable" 类还为范型哈希表定义了接口，由其余的代码使用。UsageEnvironment 中的都是抽象类；它们必须在实现中被继承。这些子类可以利用它运行的环境的特定属性，比如它的 GUI 和/或脚本环境。

BasicUsageEnvironment 库则定义了 UsageEnvironment 中的类的一个具体实现，用于简单的终端应用程序。读取事件和延迟操作使用一个 select() 循环处理。

2.groupsock

这个库中的类封装了网络接口和 sockets。特别是其中的 "Groupsock" 类封装了一个 socket，用于发送（和/或接收）组播数据报。

3.liveMedia

这个库是 live555 的核心所在。其中定义了一个类层次体系，以 "Medium" 为顶层基类，用于各种各样的流媒体类型和编解码。

4.mediaServer 和 proxyServer

mediaServer 目录下的是 "LIVE555 Media Server"，它是一个完整的 RTSP 服务器应用程

序。它可以把几种媒体文件转为流，如前面看到的，这些文件必须位于当前工作目录。这些文件包括：

- MPEG TS 文件（文件后缀名为 ".ts"）
- Matroska 或 WebM 文件（文件后缀名为 ".mkv" 或 ".webm"）
- Ogg 文件（文件后缀名为 ".ogg", "ogv" 或 ".opus"）
- MPEG-1 或 2 程序流文件（文件后缀名为 ".mpg"）
- MPEG-4 Video Elementary Stream 文件（文件后缀名为 ".m4e"）
- H.264 Video Elementary Stream 文件（文件后缀名为 ".264"）
- H.265 Video Elementary Stream 文件（文件后缀名为 ".265"）
- VOB 视频+音频文件（文件后缀名为 ".vob"）
- DV 视频文件（文件后缀名为 ".dv"）
- MPEG-1 或 2 (包括 layer III - 比如 'MP3') 音频文件（文件后缀名为 ".mp3"）
- WAV (PCM) 音频文件（文件后缀名为 ".wav"）
- AMR 音频文件（文件后缀名为 ".amr"）
- AC-3 音频文件（文件后缀名为 ".ac3"）
- AAC (ADTS 格式) 音频文件（文件后缀名为 ".aac"）

proxyServer 目录下的是 "LIVE555 Proxy Server"，它是一个单播 RTSP 服务器，它为一个或多个“后端”单播或多播 RTSP/RTP 流扮演“代理”的角色。

5.WindowsAudioInputDevice

这是 "liveMedia" 库的 "AudioInputDevice" 抽象类的一个实现。它可以被 Windows 应用程序用于从输入设备读取 PCM 音频采样。

6.testProgs

这个目录实现了一些简单的程序，使用 "BasicUsageEnvironment" 来演示如何使用这些库开发应用程序。其中除了包括用于测试库的测试应用之外，还包括 "openRTSP" - 命令行的 RTSP 客户端，"playSIP" - 命令行的 SIP 会话记录器，"vobStreamer" - 网络 DVD 播放器等工具。

2.2 RTSP 推流

基于 buildroot 系统进行测试，首先确保 buildroot 勾选以下配置，如下图所示：

```
home/topeet/Linux/rk3588-linux/buildroot/output/rockchip_rk3588/.config - Buildroot 2021.11 Configuration
->Target packages ->libraries ->Multimedia ->Search (live555)

Symbol: BR2_PACKAGE_LIVE555 [=y]
Type : bool
Prompt: live555
Location:
  -> Target packages
  -> Libraries
(1)      -> Multimedia
Defined at package/live555/Config.in:1
Depends on: BR2_INSTALL_LIBSTDCPP [=y]
```

系统启动后输入 ifconfig 查看开发板 IP，如下图所示：

```
root@topeet:/# ifconfig
eth0      Link encap:Ethernet  HWaddr EE:DC:D6:DB:A5:86
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:82

eth1      Link encap:Ethernet  HWaddr EA:DC:D6:DB:A5:86
          inet addr:192.168.1.24 Bcast:192.168.1.255 Mask:255.255.255.0
          inet6 addr: 240e:341:d020:c500:51d:cc88:ec33:f3cf/64 Scope:Global
          inet6 addr: fe80::86ca:49b1:c85b:f7a8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:721 errors:0 dropped:4 overruns:0 frame:0
          TX packets:88 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:55185 (53.8 KiB)  TX bytes:14382 (14.0 KiB)
          Interrupt:122
```

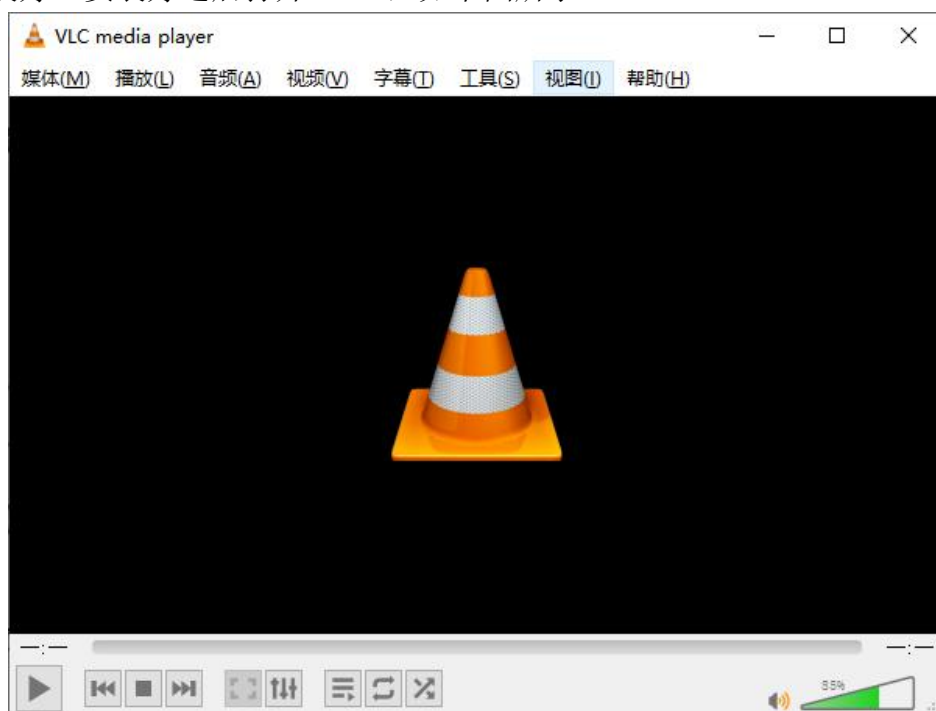
测试视频在网盘资料“iTOP-3588 开发板\02_【iTOP-RK3588 开发板】开发资料\09_Linux 系统开发配套资料\05_开发板推流配套资料\RTSP”目录下获取，拷贝到系统如下图所示：

```
root@topeet:/#
root@topeet:/# ls Fish.264
Fish.264
root@topeet:/#
```

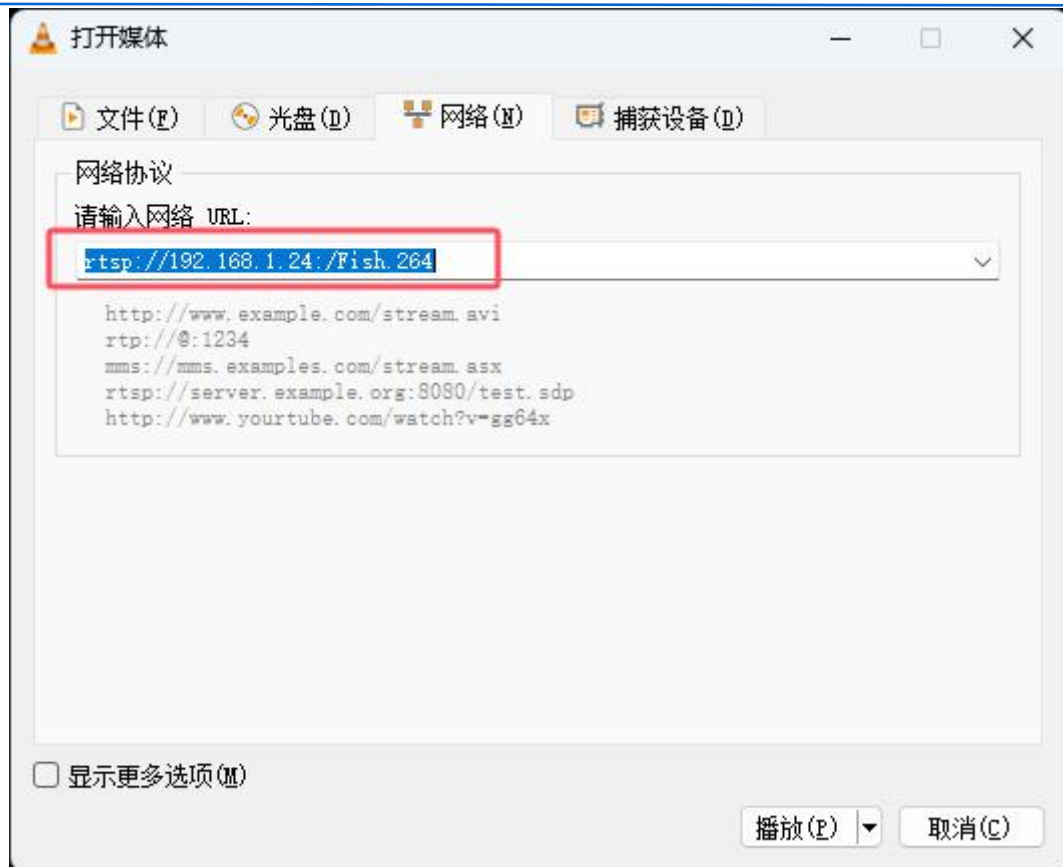
启动 live555 的流媒体服务器 live555MediaServer（在测试视频所在目录下执行），如下图所示：

```
root@topeet:/# live555MediaServer
LIVE555 Media Server
  version 1.09 (LIVE555 Streaming Media library version 2021.05.03).
Play streams from this server using the URL
  rtsp://192.168.1.24/<filename>
or
  rtsp://[240e:341:d020:c500:51d:cc88:ec33:f3cf]/<filename>
where <filename> is a file present in the current directory.
Each file's type is inferred from its name suffix:
  ".264" => a H.264 Video Elementary Stream file
  ".265" => a H.265 Video Elementary Stream file
  ".aac" => an AAC Audio (ADTS format) file
  ".ac3" => an AC-3 Audio file
  ".amr" => an AMR Audio file
  ".dv"  => a DV Video file
  ".m4e" => a MPEG-4 Video Elementary Stream file
  ".mkv" => a Matroska audio+video+(optional)subtitles file
  ".mp3" => a MPEG-1 or 2 Audio file
  ".mpg" => a MPEG-1 or 2 Program Stream (audio+video) file
  ".ogg" or ".ogv" or ".opus" => an Ogg audio and/or video file
  ".ts"  => a MPEG Transport Stream file
           (a ".tsx" index file - if present - provides server 'trick play' support)
  ".vob" => a VOB (MPEG-2 video with AC-3 audio) file
  ".wav" => a WAV Audio file
  ".webm" => a WebM audio(Vorbis)+video(VP8) file
See http://www.live555.com/mediaServer/ for additional documentation.
(We use port 80 for optional RTSP-over-HTTP tunneling.)
```

RTSP 的流媒体服务器已经成功开启，现在我们进行拉流，将 Windows 主机作为拉流端，使用 VLC 软件进行拉流，VLC 软件可以在网盘资料“[iTOP-3588 开发板\02_【iTOP-RK3588 开发板】开发资料\09_Linux 系统开发配套资料\05_开发板推流配套资料](#)”目录下找到，大家可以自行安装好。安装好之后打开 VLC，如下图所示：



点击左上角媒体->打开网络串流，弹出如下界面输入以下内容，如下图所示：



点击“播放”即可从 RTSP 流媒体服务器拉取视频数据进行播放，如下所示：

