# ZSV 8155平台编译方法

# 申请Docker

## 参考

Docker申请与登录： <https://yuque.antfin-inc.com/shangbeibei.bbs/oiq6bt/lgoa24>

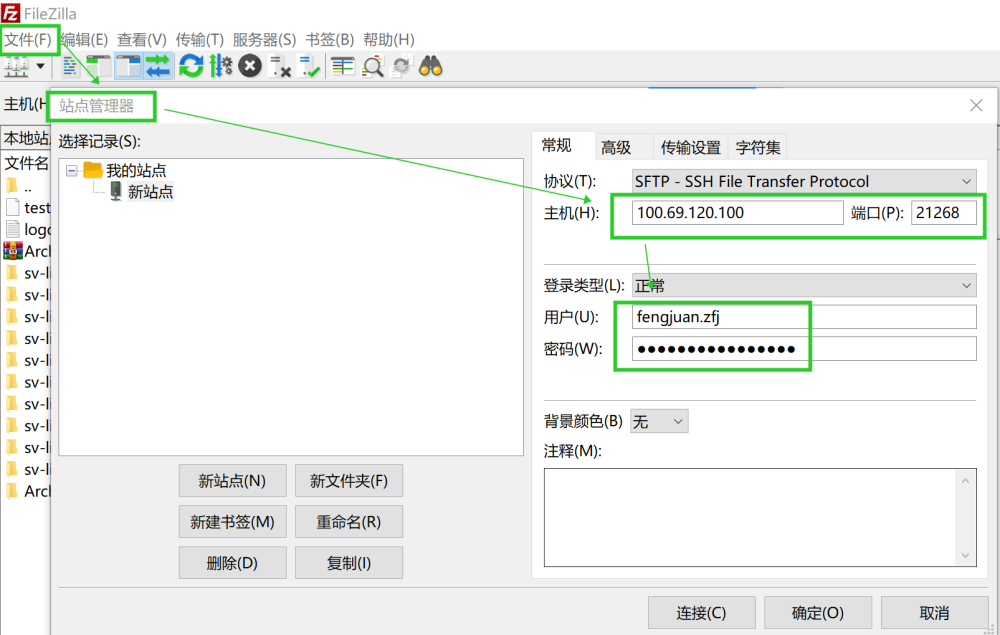
vscode 登录docker配置：<https://blog.csdn.net/zfjBIT/article/details/123095065?spm=1001.2014.3001.5501>

# Docker 与 本地文件传输工具

filezilla 客户端： <https://www.filezilla.cn/download/client>



文件-站点管理：



# 交叉编译环境配置

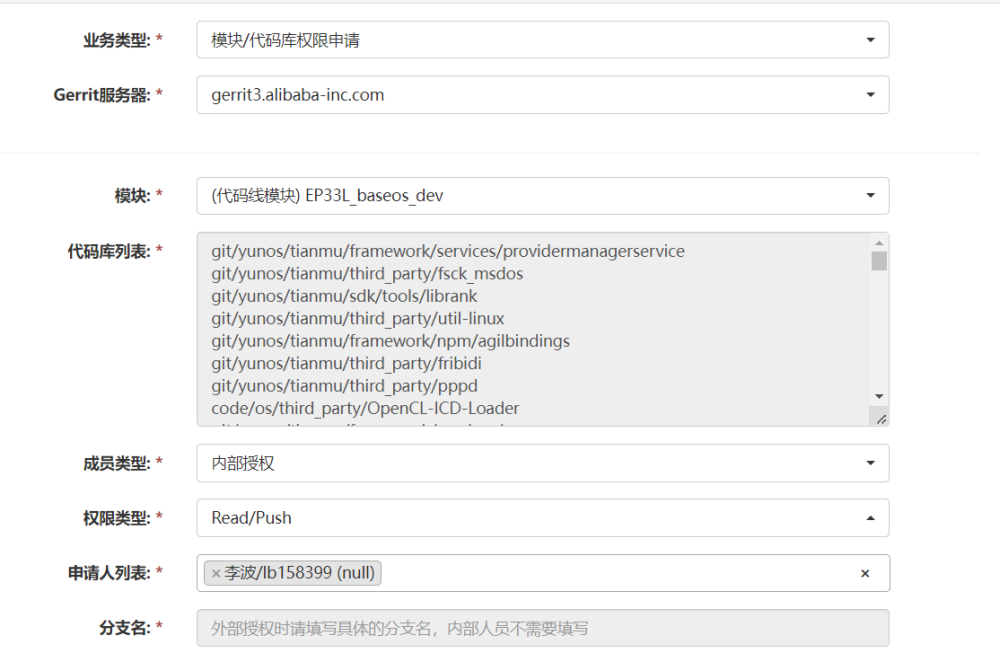
EP33L\_baseos\_dev、AS33\_baseos\_dev\_ivi 平台可通用，编译出来的库均可在8155调用；

## EP33L\_baseos\_dev交叉编译环境配置

### 参考：使用文档中的方法1

<https://yuque.antfin-inc.com/docs/share/2558e491-ce3a-4958-bd2a-29a0b9aa0880?#>

1. ~~申请全仓代码~~ [~~https://scm.yunos-inc.com/gerrit\_task~~](https://scm.yunos-inc.com/gerrit_task)



更新：参考<https://aliyuque.antfin.com/uwk7mr/lvhh2l/gwhn8c>申请代码分支2. 下载源码

cd workspace

mkdir EP33L\_baseos\_dev

cd EP33L\_baseos\_dev

repo init -u ssh://gerrit3.alibaba-inc.com:29418/repo/banma/codebase -b EP33L\_baseos\_devrepo sync -c --no-clone-bundle -j48 --no-tags

1. 编译zsv算法依赖

（1）编译 sa8155 user 系统

cd xmake && source ./xmake.sh --with-packagelevel=ic\_core --enable-multicore --with-targettype=ic

选择sa8155 user

find: ‘vendor/qcom/sa8155/voice\_localdialog/asr’: No such file or directory  
find: ‘vendor/qcom/sa8155/voice\_localdialog/nlu’: No such file or directory  
find: ‘vendor/qcom/sa8155/voice\_localdialog/dm/xml’: No such file or directory  
xmake/platform/product.mk:73: include vendor/qcom/custom/common/product.mk  
XMAKE SUCCESSFULLY!!!   
Source x successful.

打印以上信息，表示编译完成；

（2）编译zsv依赖项：

#EP33L\_baseos\_dev目录下cd third\_party/glib/ && mma -j3cd vendor/qcom/sa8155/yunhal/gpu && mmcd third\_party/zlib/ && mma -j3cd third\_party/openssl/ && mma -j3cd third\_party/libjpeg-turbo/ && mm -j3cd third\_party/libpng/ && mm -j3cd third\_party/openh264/ && mm -j3cd third\_party/libwebp && mm -j3cd third\_party/libpthread-stubs && mma -j3cd third\_party/ffmpeg && mma -j3cd third\_party/opencv && mm -j3cd framework/libs/base/log && mm -j3cd framework/libs/algo && mm

（3）建立软连接

#EP33L\_baseos\_dev目录下  
ln -s out/platforms/sa8155.user.ic out/platforms/sa8155.user.ivi

### 可能报错

openh264、opencv编译报错

cd third\_party/openh264/ && mm -j3  
cd third\_party/opencv && mm -j3

解决办法：

使用mma编译即可成功；

cd third\_party/openh264/ && mma -j3  
cd third\_party/opencv && mma -j3

### ZSV-AS33源码交叉编译

#### 源码下载

下载AS33\_dev 分支：

git clone -b zsv\_AS33 http://gitlab.alibaba-inc.com/BANMA\_SURROUND\_VIEW/zsv.git zsv\_AS33

#### 修改toolchain.cmake

zsv\_AS33/platform/sa8155/toolchain.cmake

修改TOOLCHAIN\_ROOT\_DIR、ROOTFS\_DIR为本地路径：

#set(TOOLCHAIN\_ROOT\_DIR "/home/lb158399/workspace/SA8155/EP33\_213\_SOP")  
#set(ROOTFS\_DIR "/home/lb158399/workspace/SA8155/EP33\_213\_SOP/out/platforms/sa8155.user.ivi/target/rootfs")  
  
set(TOOLCHAIN\_ROOT\_DIR "/home/fengjuan.zfj/workspace/EP33L\_baseos\_dev")  
set(ROOTFS\_DIR "/home/fengjuan.zfj/workspace/EP33L\_baseos\_dev/out/platforms/sa8155.user.ivi/target/rootfs")

#### 编译

cd platform/sa8155  
source build.opencv  
source build.zsv

【2022.06.14更新】

工程系统配置有opencv，所以无需再编译opencv，即无需~~source build.opencv~~

使用系统opencv：

头文件路径：${TOOLCHAIN\_ROOT\_DIR}/ZXQ/workspace/SrvService/3rdparty/algorithm/zsv/include/zsv/cv/

库路径：${TOOLCHAIN\_ROOT\_DIR}/out/platforms/sa8155.user.ivi/target/rootfs/usr/lib64/

注意：

目前${TOOLCHAIN\_ROOT\_DIR}/out/platforms/sa8155.user.ivi/target/rootfs/usr/lib64/的opencv相关库是cd third\_party/opencv && mm -j3编译出来的，版本是3.4.5；

头文件版本号是3.2.0；

暂无不同版本相关接口定义不一致等相关兼容问题报错；

无需配置库目录，交叉编译时会查找${TOOLCHAIN\_ROOT\_DIR}/out/platforms/sa8155.user.ivi/target/rootfs/usr/lib64/；

需修改opencv 头文件目录，如下：

修改 platform/sa8155/cmakes/opencv.cmake

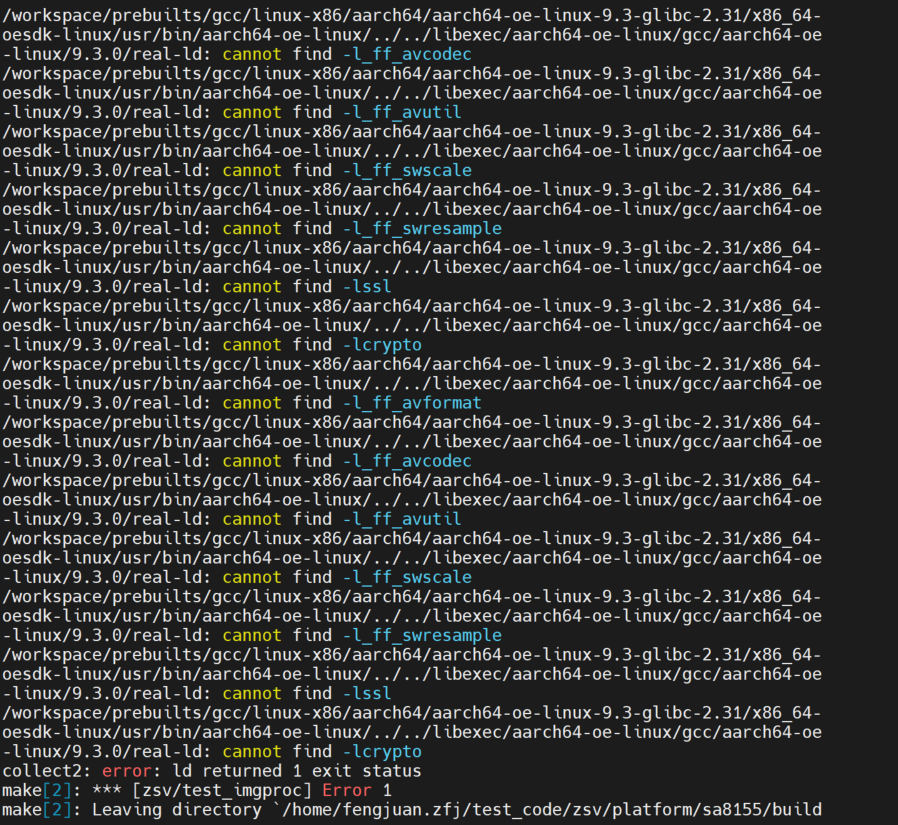
#set(OpenCV\_INCLUDE\_DIRS "${PROJECT\_SOURCE\_DIR}/opencv\_ext/include/")  
set(OpenCV\_INCLUDE\_DIRS "${TOOLCHAIN\_ROOT\_DIR}/ZXQ/workspace/SrvService/3rdparty/algorithm/zsv/include/zsv/cv/")

执行编译：

cd platform/sa8155  
source build.zsv

#### 可能报错

source build.zsv 执行到36%时报如下错误：



ffmpeg相关错误，之前环境配置时ffmpeg模块安装并未报错，确认确实安装了相关库，检查sa8155.user.ivi的权限：

解决办法：

删掉之前建立的超链接；

cd out/platforms/  
rm -rf sa8155.user.ivi

修改权限后重新建立超链接：

chmod 777 sa8155.user.ic/  
ln -s sa8155.user.ic sa8155.user.ivi

### ZSV-EP33L源码交叉编译

流程大致与上述AS33源码编译相同，唯一区别在于cdsp优化库部分，这里着重介绍。

#### 背景

EP33L项目中为了降低算法cpu占用率引入了朱天阳团队的优化代码，包括neon优化/cdsp异构计算等方式的实现接口。相关代码由天阳团队维护，目前未放置于ZSV算法文件中。

#### 功能

目前天阳团队提供的优化功能包括：1）使用cdsp运行单视图畸变矫正；2）使用cdsp运行环视拼接；3）使用neon加速透明底盘中的双三次图像插值

#### 开启方式

在platform\sa8155\CMakeLists.txt中进行相关的开关控制

1）27行中将ENABLE\_ALG\_OPT设为ON，此时将在编译中引入优化相关的库与头文件

2）48-50行中的三个宏定义分别控制上述功能中的三个对应优化执行开关，目前根据工程需求打开1）和3）

#### 编译相关

对于工程侧同学：docker中的系统分支优化代码由天阳团队维护更新，无其他操作直接编译即可

对于算法侧同学：docker中的系统分支优化代码可能是老版本，若出现编译错误，可将platform\sa8155\cdsp\include 文件夹下所有文件拷贝到你的docker环境/ framework/libs/algo/include目录下，将platform\sa8155\cdsp\lib下.so文件拷贝到你的docker环境/ out/platforms/sa8155.user.ic/target/rootfs/lib64目录下再执行编译；由于platform\sa8155\cdsp文件夹并不会更新，这里只是为了编译通过

## AS33\_baseos\_dev\_ivi交叉编译环境配置（失败）

系统源码下载：

repo init -u ssh://gerrit3.alibaba-inc.com:29418/repo/banma/codebase -b AS33\_baseos\_dev\_ivi

repo sync

编译：

cd xmake

change\_node.sh v8.10.0 && source xmake.sh sa8155\_user && xmake

【编译失败==============】