arm_soc平台编译准备

1 概述

arm SoC平台，内部已经集成了相应的libsophon、sophon-opencv和sophon-ffmpeg运行库包，位于`/opt/sophon/`下。

通常在x86主机上交叉编译程序，使之能够在arm SoC平台运行。您需要在x86主机上使用SOPHON SDK搭建交叉编译环境，将程序所依赖的头文件和库文件打包至soc-sdk目录中。

2 编译环境准备

本章节需要提前准备libsophon-soc和sophon-mw包，请联系技术支持获取。

2.1 安装交叉编译工具链

```bash
sudo apt-get install gcc-aarch64-linux-gnu g++-aarch64-linux-gnu
```

2.2 准备libsophon

```bash
# 创建依赖文件的根目录
mkdir -p soc-sdk
# 解压sophon-img release包里的libsophon_soc_${x.y.z}_aarch64.tar.gz，其中x.y.z为版本号
tar -zxf libsophon_soc_${x.y.z}_aarch64.tar.gz
# 将相关的库目录和头文件目录拷贝到依赖文件根目录下
cp -rf libsophon_soc_${x.y.z}_aarch64/opt/sophon/libsophon-${x.y.z}/lib ${soc-sdk}
cp -rf libsophon_soc_${x.y.z}_aarch64/opt/sophon/libsophon-${x.y.z}/include ${soc-sdk}
```

2.3 准备ffmpeg和opencv

```bash
# 解压sophon-mw包里的sophon-mw-soc_${x.y.z}_aarch64.tar.gz，其中x.y.z为版本号
tar -zxf sophon-mw-soc_${x.y.z}_aarch64.tar.gz
# 将ffmpeg和opencv的库目录和头文件目录拷贝到依赖文件根目录下
cp -rf sophon-mw-soc_${x.y.z}_aarch64/opt/sophon/sophon-ffmpeg_${x.y.z}/lib ${soc-sdk}
cp -rf sophon-mw-soc_${x.y.z}_aarch64/opt/sophon/sophon-ffmpeg_${x.y.z}/include ${soc-sdk}
cp -rf sophon-mw-soc_${x.y.z}_aarch64/opt/sophon/sophon-opencv_${x.y.z}/lib ${soc-sdk}
cp -rf sophon-mw-soc_${x.y.z}_aarch64/opt/sophon/sophon-opencv_${x.y.z}/include ${soc-sdk}
```

2.4 准备第三方库

依赖libeigen3-dev、libgflags-dev、libgoogle-glog-dev、libexiv2-dev

2.4.1 准备和构建qemu虚拟环境

```bash
# 安装qemu
sudo apt-get install -y qemu-user-static debootstrap
# 创建文件夹，并构建虚拟环境，映射到rootfs文件夹内
mkdir rootfs
cd rootfs
# 构建 ubuntu 20.04的rootfs
sudo qemu-debootstrap --arch=arm64 focal .
sudo chroot . qemu-aarch64-static /bin/bash

# 进入qemu 后，安装libeigen3-dev、libgflags-dev、libgoogle-glog-dev、libexiv2-dev
```

```
apt-get install -y software-properties-common
apt-add-repository universe
apt-get update
apt-get install -y libeigen3-dev libgflags-dev libgoogle-glog-dev libexiv2-dev

# 使用exit命令，退出qemu虚拟环境
exit
```

2.4.2 拷贝第三方库的头文件和库

```bash
# 退出qemu虚拟环境后
# libgoogle-glog-dev
cp -rf ${rootfs}/usr/lib/aarch64-linux-gnu/libglog* ${soc-sdk}/lib
cp -rf ${rootfs}/usr/include/glog ${soc-sdk}/include
# libgflags-dev
cp -rf ${rootfs}/usr/lib/aarch64-linux-gnu/libgflags* ${soc-sdk}/lib
cp -rf ${rootfs}/usr/include/gflags ${soc-sdk}/include
# libexiv2-dev
cp -rf ${rootfs}/usr/lib/aarch64-linux-gnu/libexiv2* ${soc-sdk}/lib
cp -rf ${rootfs}/usr/include/exiv2 ${soc-sdk}/include
# libeigen3-dev
cp -rf ${rootfs}/usr/include/eigen3 ${soc-sdk}/include
```

> 这里，交叉编译环境和相关依赖环境的准备步骤已经准备完成，接下来可以编译需要在SoC平台上运行的程序。