**J5编译环境搭建**

**背景**

我们在做J5开发时，有需要外部伙伴为我们提供J5平台的SDK的情形。那么，外部伙伴怎样才能为我们提供J5平台可用的SDK呢？或者说，**外部伙伴如何J5编译**？

思路有两个：

* **CMake交叉编译法**：轻舟提供基于CMake的交叉编译工具链和SysRoot 给对方（集成到交叉编译Docker镜像）。
* 优点：编译更快速
* 缺点1：由于轻舟内部的交叉编译环境配置复杂，且使用的是bazel编译体系，而非CMake。所以适配起来难度较大
* 缺点2：交叉编译的binary无法在x86环境下运行，需要放在台架上测试。
* **Qemu Native编译法**：在此方法中，我们提供兼容J5 原生环境的AArch64的Docker镜像（可称为**J5兼容Docker镜像**），镜像中提前安装好一些主要的第三方库并固定其版本。而我们的J5 交叉编译SysRoot 也是从这个Docker镜像派生而来。这样，合作伙伴有两种选择：
* 1）在x86\_64宿主机上，通过Qemu 启动**J5兼容Docker镜像**，从而实现在x86\_64上也能编译构建（甚至运行）J5 AArch64的SDK及其示例程序。当然，因为涉及Qemu, 缺点是略慢一点。
* 2）如果用户有AArch64 的宿主机，例如xavier等支持docker环境的Linux，则无需借助Qemu 也可启动**J5兼容Docker镜像**，就不慢了。

因此我们选择法二：**Qemu Native 编译法。**

**J5兼容Docker镜像的启动**

我们以x86平台的Ubuntu 18.04为例。

**安装docker**

可以参考这个文档：https://www.myfreax.com/how-to-install-and-use-docker-on-ubuntu-18-04/

**设置Qemu**

在x86\_64宿主机上，需设置Qemu（一次性设置，机器重启后失效）。注意在AArch64宿主机上不需要这一步。

|  |
| --- |
| Bash  docker run --rm --privileged multiarch/qemu-user-static --reset -p yes |

**加载j5编译镜像**

下载这个镜像文件：

**[j5-qemu.tar.gz]**

完成之后执行如下命令：

|  |
| --- |
| Plain Text docker load < j5-qemu.tar.gz |

该镜像是基于arm64v8/ubuntu:18.04镜像 而来，其中的GCC 默认版本为9.x, CMake 为3.25.1, OpenCV 为 4.5.5。

其中还集成了其它常见的package，具体参考这个Dockerfile（省略了opencv的安装步骤）：

|  |
| --- |
| Plain Text FROM arm64v8/ubuntu:18.04  RUN apt-get update && apt-get -y install --no-install-recommends \  curl \  wget \  gnupg2 \  python3 \  python3-pip \  python3-setuptools \  software-properties-common \  && apt-get clean \  && rm -rf /var/lib/apt/lists/\*   RUN sed -i.bak 's;bionic;focal;g' /etc/apt/sources.list \  && apt-get update && apt-get -y install --no-install-recommends \  gcc-9 \  g++-9 \  libstdc++-9-dev \  libc6-dev \  && update-alternatives --install /usr/bin/gcc gcc /usr/bin/gcc-9 90 \  && update-alternatives --install /usr/bin/g++ g++ /usr/bin/g++-9 90 \  && mv /etc/apt/sources.list.bak /etc/apt/sources.list \  && apt-get clean \  && rm -rf /var/lib/apt/lists/\*  RUN apt-get update && apt-get -y install --no-install-recommends \  coinor-libipopt-dev \  libacl1-dev \  libasound2-dev \  libpcap-dev \  libssl-dev \  libopenblas-dev \  libusb-1.0-0-dev \  libmp3lame-dev \  libfftw3-dev \  libudev-dev \  zlib1g-dev \  libdouble-conversion-dev \  libsnappy-dev \  libglu1-mesa-dev \  && apt-get clean \  && rm -rf /var/lib/apt/lists/\* |

除此之外，地图常用的libproj和libsqlite3也都是预先安装好的，且版本比apt安装的版本要高一些。建议除非出现了编译时错误（提示依赖不存在），否则不建议额外安装这些library。

如果合作伙伴还需要其他的依赖库，可以自行进入docker环境里面安装（进入docker环境的方式见下文）。

为避免第三方依赖库的版本冲突，建议合作伙伴的第三方依赖尽可能通过APT安装。另，可参考 <https://gcc.gnu.org/wiki/Visibility> 来处理好 SDK 的shared library visibility 的问题。

**启动和进入J5兼容Docker镜像**

**启动**

Note：为了编译代码，我们需要把代码目录映射进docker环境。假定我们的代码目录是当前目录下的sdk目录，那么启动docker命令的时候可以加一个参数-v sdk:sdk。完整的命令如下（注意其中的sdk需要改成实际名称）：

|  |
| --- |
| Bash docker run -itd --name j5 -v sdk:/sdk registry.qcraftai.com/global/qcraft-dev:j5-bionic-20221230\_0010 |

**进入**

|  |
| --- |
| Bash docker exec -it j5 /bin/bash |

如果一切正常，我们就进入了一个J5的运行时环境。我们可以执行如下命令：

1. 查看系统信息（可以看到运行时环境已经不是x86，而是aarch64了。我们可以在里面执行aarch64程序，无需台架）：

|  |
| --- |
| Plain Text root@5b68a293e206:/# uname -a Linux 5b68a293e206 5.4.0-72-generic #80~18.04.1-Ubuntu SMP Mon Apr 12 23:26:25 UTC 2021 aarch64 aarch64 aarch64 GNU/Linux |

1. 查看关键程序的版本：

|  |
| --- |
| Plain Text root@5b68a293e206:/# g++ --version g++ (Ubuntu 9.4.0-1ubuntu1~20.04.1) 9.4.0 Copyright (C) 2019 Free Software Foundation, Inc. This is free software; see the source for copying conditions. There is NO warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  root@5b68a293e206:/# cmake --version cmake version 3.25.1  CMake suite maintained and supported by Kitware (kitware.com/cmake). |

1. 进入代码目录：由于在”启动“时，我们把代码映射到了/sdk，所以可以直接cd /sdk查看代码并执行编译、运行等操作。
2. 安装额外的依赖：直接使用apt即可（以libpcap-dev为例）

|  |
| --- |
| Plain Text apt update apt install libpcap-dev |

**备注**：如果自行安装了依赖库，请及时反馈给轻舟。这样轻舟可以在硬件J5上同步添加这些依赖。