Rockchip Buildroot 集成编译 ROS2 说明

文件标识：RK-SM-YF-912

发布版本：V1.0.0

日期：2021-09-09

文件密级：□绝密 □秘密 □内部资料 ■公开

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

RK Buildroot SDK 集成了ROS1的编译，即对应的defconfig引用ros_indigo.config 或ros_kinetic.config。 针对ROS2，Buildroot仅提供ROS2所需要的依赖包，将ROS2编译独立出来放到Docker中去编译。这样做的目的是将ROS2与Buildroot彻底分开，方便ROS2版本更新维护。

编译ROS2分为两个步骤：

- 首先完成Buildroot rootfs编译，它包含了一些ROS2所需要的应用包，比如python3、bullet、opencv、eigen等
- 进入Docker，使用Buildroot rootfs的交叉工具链直接编译ROS2

当前已经验证可编译通过的ROS2有foxy, galactic两个版本。

产品版本

芯片名称	内核版本
RK356x	Kernel 4.19

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

修订记录

版本号	作者	修改日期	修改说明
V1.0.0	zhengsq	2021-09-09	初始版本

目录

1. 使用Docker编译所需的文件及说明

如下文件是Docker编译ROS2所需要的，他们位于Linux SDK的`buildroot/package/rockchip/ros2`目录下。

```
tree -L 1 buildroot/package/rockchip/ros2
.
├── build_ros2.sh      # 主要的编译脚本
├── mixins             # 编译环境参数，只设置了aarch64, arm未配置
├── rosdep.Dockerfile        # Docker image基础包配置文件
└── sysroot.Dockerfile       # ROS2编译的Docker image配置文件，包含依赖环境及源码下载

```

2. Buildroot rootfs编译ROS2的依赖包

Buildroot 应该更新到包含ros2_dep.config的版本，例如：

```
cd buildroot
git log --oneline -4
50d87c26e7 configs: rockchip: add ros2 build dependencies
397ee958e8 package/python-packaging: new package with version 20.9
e9dae80ec5 package/python-six: bump to version 1.15.0
8aa73ca79c package/python-pyparsing: bump to version 2.4.7

```

ros2_dep.config提供了编译、运行ROS2所需要的应用包，需要添加并编译到rootfs。 在Buildroot defconfig中，通过引用ros2_dep.config添加依赖关系。示例如下：

```
git diff
--- a/configs/rockchip_rk356x_robot_defconfig
+++ b/configs/rockchip_rk356x_robot_defconfig
@@ -10,6 +10,7 @@
 #include "wifi.config"
 #include "debug.config"
 #include "bt.config"
+#include "ros2_dep.config"
 BR2_TARGET_GENERIC_HOSTNAME="rk356x_robot"
 BR2_TARGET_GENERIC_ISSUE="Welcome to RK356X Buildroot For Robot"
 BR2_ROOTFS_OVERLAY:="board/rockchip/common/robot/base board/rockchip/common/wifi"

```

完整编译rootfs后，可进入下一步。

3. 准备Linux(Ubuntu) PC机的依赖环境

ROS2文档[1][2]显示，它提供了Docker镜像用以交叉编译arm/aarch64，比如Dockerfile_ubuntu_arm64_prebuilt[2]。但目标系统仅支持ubuntu 18.04(bionic)，与RK Linux SDK中的rootfs相差较多。借鉴Dockerfile_ubuntu_arm64_prebuilt，我们提供另一Dockerfile以适配RK Linux SDK用于交叉编译ROS2。RK Linux SDK使用了较新的gcc版本，比如rk356x使用了gcc9.3，交叉工具链的个别包会依赖于较新版本的GLIBC，比如GLIBC_2.29。因此在选择Docker镜像时，提供的Docker镜像是基于ubuntu 20.04(focal)。

Ubuntu PC机上安装docker程序：

```
sudo apt install docker.io
sudo usermod -aG docker $USER
newgrp docker  # 登录docker用户组

```

(可选)修改/etc/docker/daemon.conf镜像源，根据所在网络情况，比如改成阿里、腾讯等：

```
cat /etc/docker/daemon.json
{"registry-mirrors": ["https://registry.cn-hangzhou.aliyuncs.com"]}

sudo systemctl daemon-reload
sudo systemctl restart docker

```

3.1 网络需求

ROS2源码大部分从github下载，需要有较稳定的github访问的网络。建议最好能有VPN，否则整个下载及编译可能很难完成。

4. 使用Docker编译ROS2

选择Docker镜像时，以下2点会便于ROS2的编译：

- 使得Buildroot的交叉工具链版本与Ubuntu 相匹配
- Docker中的python版本与Buildroot sdk中保持一致，比如都是3.8

下载原始的Ubuntu Docker镜像rosdep：

```
docker build -t rosdep -f rosdep.Dockerfile ./  # "./"不要少拷贝了，表示当前目录

```

如果上述**docker build**执行失败，提示网络出错、**connection reset by peer**等，可尝试换一个网络、设置**DNS**为**8.8.8.8**，关闭防火墙，甚至尝试下手机**5G**网络。

基于rosdep，添加ROS2编译的环境并且下载ROS2源码：

```
docker build --build-arg BASE_IMAGE=rosdep \
    --build-arg ROS_VERSION=ros2 \
    --build-arg ROS_DISTRO=foxy \
    -t ros_sysroot -f sysroot.Dockerfile ./

```

说明：

- ROS_VERSION只能是ros2
- ROS_DISTRO表示具体的版本[3]，其中已验证foxy及galactic
- BASE_IMAGE参数rosdep表示基础的Ubuntu 镜像名称

Docker Image(sys_rootfs)在创建时已经指定了Entrypoint为`build_ros2.sh`，可直接执行编译如下：

```
OUTPUT=/data/linux-sdk/rk3566/buildroot/output/rockchip_rk356x_robot
docker run --rm -it \
    --mount type=bind,source=${OUTPUT},target=/buildroot \
    ros_sysroot

# 编译成功后，应有类似提示：
...
Summary: 276 packages finished [15min 37s]
...
build ros quit & cleanup

```

说明：

- 编译过程主要定义在build_ros2.sh
- 命令中/data/linux-sdk/rk3566/buildroot/output/rockchip_rk356x_robot请替换成相应SDK的路径
- 编译生成的目标文件位于${OUTPUT}/target/opt/ros目录下

如build_ros2.sh未提示错误即成功编译。其中，还有部分包在Buildroot SDK环境中，无法编译、执行的，比如：

- rviz，依赖于X11/desktop。如果你需要这个功能，直接使用Ubuntu arm镜像，而不是Buildroot
- turtlesim，依赖于QT5，在部分产品上（如扫地机类），Buildroot未编译QT5
- 如果想要取消某个包的编译，在src对应的路径下，创建一个COLCON_IGNORE即可。比如`touch src/ros/ros_tutorials/turtlesim/COLCON_IGNORE`

4.1 修改ROS2编译选项(可选）

如果需要修改编译ROS2选项，例如取消部分应用包编译、增加demo程序、排查错误等，在`docker run`命令中使用`--entrypoint bash`选项进入命令行，然后执行`./build_ros2.sh`编译脚本。

```
OUTPUT=/data/linux-sdk/rk3566/buildroot/output/rockchip_rk356x_robot
docker run --rm -it --entrypoint /bin/bash \
    --mount type=bind,source=${OUTPUT},target=/buildroot \
    ros_sysroot

# 现在已经进入Docker
root@97eacf55d026:/opt/ros/foxy# ls
build_ros2.sh  ros2.repos  src

root@97eacf55d026:/opt/ros/foxy# ./build_ros2.sh

```

4.2 单独编译某个ROS2 package 及应用程序

使用colcon build的参数`--packages-select <pacage_name>`可单独编译包，详情可参考`./build_ros2.sh`脚本。

编译您自己的应用程序，可自行写cmake，使用buildroot的交叉工具链，引用相应的头文件、动态链接库即可。如您对ROS2的ament比较熟悉，也可在ament的基础上写cmake。ROS2源码中已经包含了许多demo程序，请先参考。

5. 打包rootfs并运行ROS2

在上述ROS2完整编译结束后，进入到buildroot sdk，重新打包rootfs即可。ROS2安装在/opt/ros目录下。

前面Docker默认用户是root，因此将ROS2安装的文件属性也会是root，但不影响打包rootfs.img。

```
cd /data/linux-sdk/rk3566
./build.sh rootfs    # 重新打包rootfs.img

```

烧录rootfs.img后，进入rk3566板端，执行Hello World Demo：

```
# cd /opt/ros/
# export COLCON_CURRENT_PREFIX=/opt/ros
# source ./local_setup.sh
# ros2 pkg list
# ros2 pkg executables

# ros2 run demo_nodes_cpp listener &
# ros2 run demo_nodes_cpp talker
[INFO] [1501839280.834017748] [talker]: Publishing: 'Hello World: 1'
[INFO] [1501839280.839280957] [listener]: I heard: [Hello World: 1]
[INFO] [1501839281.831636015] [talker]: Publishing: 'Hello World: 2'
[INFO] [1501839281.835092640] [listener]: I heard: [Hello World: 2]
[INFO] [1501839282.831618532] [talker]: Publishing: 'Hello World: 3'
[INFO] [1501839282.835336782] [listener]: I heard: [Hello World: 3]

# ros2 run demo_nodes_py listener &
# ros2 run demo_nodes_py talker
```

6. 参考索引

1. https://docs.ros.org/en/foxy/Guides/Cross-compilation.html
2. https://github.com/ros-tooling/cross_compile.git
3. https://docs.ros.org/en/foxy/Releases.html