Rockchip ROS2 Introduction

ID: RK-SM-YF-912

Release Version: V1.0.0

Release Date: 2021-09-09

Security Level: □Top-Secret □Secret □Internal ■Public

Rockchip Electronics Co., Ltd.

No.18 Building, A District, No.89, software Boulevard Fuzhou, Fujian,PRC

Website: www.rock-chips.com

Customer service Tel: +86-4007-700-590

Customer service Fax: +86-591-83951833

Customer service e-Mail: fae@rock-chips.com

Preface

Overview

RK Buildroot SDK integrates ROS1 compilation, that is, the corresponding defconfig references ros_indigo.config or ros_kinetic.config.
For ROS2, Buildroot only provides the dependency packages needed by ROS2, and ROS2 is compiled in Docker independently, aiming to separate ROS2 from Buildroot completely and facilitate ROS2 version update and maintenance.

To compile ROS2, there will be two steps:

- Firstly, you have to finish Buildroot rootfs compilation, which contains some application packages required by ROS2, such as python3, bullet, opencv, eigen, etc.
- Secondly, enter Docker, compile ROS2 directly by the cross tool chain of Buildroot rootfs

There are two verified versions of ROS2: foxy and galactic, both can be compiled successfully.

Product Version

Chipset	Kernel Version
RK356x	Kernel 4.19

Intended Audience

This document (this guide) is mainly intended for:

Technical support engineers

Software development engineers

Revision History

Version	Author	Date	Change Description
V1.0.0	zhengsq	2021-09-09	Initial version

Contents

1. The Required Files of Compiling ROS2 in Docker

The following files are needed when compiling ROS2 in Docker. They are located in the `buildroot/package/rockchip/ros2` directory of the Linux SDK.

```
tree -L 1 buildroot/package/rockchip/ros2
.
├── build_ros2.sh # The major compilation script
├── mixins # Compilation environment parameters, only aarch64 is configured, arm is not configured
├── rosdep.Dockerfile # The basic package configuration file of Docker image
└── sysroot.Dockerfile # The Docker image configuration file compiled by ROS2, including dependent environment and source code download
```

2. Dependencies of Compiling ROS2 by Buildroot rootfs

Buildroot should be updated to the version that includes ros2_dep.config, for example:

```
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
```

The ros2_dep.config provides the application packages needed to compile and run ROS2, which need to be added and compiled to rootfs. In Buildroot defconfig, add dependencies by referencing ros2_dep.config. Examples are as follows:

```
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"
```

After the rootfs is fully compiled, you can continue to the following steps..

3. Linux (Ubuntu) PC Environment Preparation

The ROS2 document [1][2] shows that it provides a Docker image for cross-compiling arm/aarch64, such as Dockerfile_ubuntu_arm64_prebuilt[2]. But the target system only supports ubuntu 18.04 (bionic), which is quite different from the rootfs in the RK Linux SDK. Learning from Dockerfile_ubuntu_arm64_prebuilt, we provide another Dockerfile to adapt to the RK Linux SDK for cross-compiling ROS2. RK Linux SDK uses a newer gcc version, for example, rk356x uses gcc9.3, and individual packages of the cross tool chain will depend on the newer version of GLIBC, such as GLIBC_2.29. Therefore, when choosing a Docker image, the provided Docker image is based on ubuntu 20.04 (focal).

Install the Docker program in a Ubuntu PC:

```
sudo apt install docker.io
sudo usermod -aG docker $USER
newgrp docker  # Log in to docker user group
```

(Optional) Modify the /etc/docker/daemon.conf image source according to the network situation, for example, change it to Alibaba, Tencent, etc.:

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

sudo systemctl daemon-reload
sudo systemctl restart docker
```

3.1 Network Requirements

Most of the ROS2 source code is downloaded from github, so a stable github network is required. It is recommended to have a VPN, otherwise the download and compilation may be difficult to complete.

4. Using Docker to Compile ROS2

When choosing a Docker image, the following 2 points will help for compilation of ROS2:

- Match the cross toolchain version of Buildroot with Ubuntu
- The python version in Docker is consistent with the Buildroot sdk, for example, both are 3.8

Download the original Ubuntu Docker image rosdep:

```
docker build -t rosdep -f rosdep.Dockerfile ./  # Don’t forget the "./", it means
the current directory
```

If the above Docker build fails, it may be related to a network error, connection reset by peer, etc., you can try to change the network, set the DNS to 8.8.8.8, turn off the firewall, or even try the mobile phone 5G network.

Based on rosdep, add the ROS2 compilation environment and download the ROS2 source code:

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

Note:

- ROS_VERSION can only be ros2
- ROS_DISTRO represents the specific version [3], in which foxy and galactic have been verified
- The BASE_IMAGE parameter rosdep represents the name of the basic Ubuntu image

Docker Image (sys_rootfs) has specified Entrypoint as `build_ros2.sh` when it is created, and it can be compiled directly as follows:

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

# After the compilation is successful, there will be a prompt similar to:
...
Summary: 276 packages finished [15min 37s]
...
build ros quit & cleanup
```

Note:

- The compilation process is mainly defined in build_ros2.sh
- Please replace /data/linux-sdk/rk3566/buildroot/output/rockchip_rk356x_robot with the path of the corresponding SDK in the command
- The compiled target file is located in the ${OUTPUT}/target/opt/ros directory

If build_ros2.sh does not prompt an error, it compiles successfully. There are some packages that cannot be compiled and executed in the Buildroot SDK environment, such as:

- rviz, depends on X11/desktop. If you need this feature, use the Ubuntu arm image directly instead of Buildroot
- turtlesim, depends on QT5, there are some products (such as sweepers), Buildroot does not compile QT5
- If you want to cancel the compilation of a certain package, just create a COLCON_IGNORE file in the path corresponding to src. For example, `touch src/ros/ros_tutorials/turtlesim/COLCON_IGNORE`

4.1 ROS2 Compilation Options(Optional)

If you need to modify the ROS2 compilation options, such as canceling part of the application package compilation, adding demo programs, troubleshootings, etc., use the `--entrypoint bash` option in the `docker run` command to enter the command line, and then execute `./build_ros2.sh` to execute the script.

```
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

# It is entering Docker now
root@97eacf55d026:/opt/ros/foxy# ls
build_ros2.sh  ros2.repos  src

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

4.2 Compile a ROS2 Package and Application Separately

Use the colcon build parameter `--packages-select <pacage_name>` to compile a package separately. Please refer to the `./build_ros2.sh` script for details, .

To compile your own application, you can write a cmake by yourself, use the cross tool chain of buildroot, and reference the corresponding header files and shared libraries. If you are familiar with ament of ROS2, you can also write a cmake base on ament. Many demo programs have been included in the ROS2 source code, please refer to them first.

5. Package rootfs and Run ROS2

When finishing the above ROS2 compilation, enter the buildroot sdk and re-package the rootfs. ROS2 is installed in the /opt/ros directory.

The default user of above Docker is root, so the file attribute for installing ROS2 will also be root, but it does not affect the packaging of rootfs.img.

```
cd /data/linux-sdk/rk3566
./build.sh rootfs   # Re-package rootfs.img
```

After flashing rootfs.img, enter the rk3566 board and execute 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. Reference Index

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