OpenCV4 使用手册 V1.0

-正点原子 ATK-DLRK3568

正点原子公司名称 ：广州市星翼电子科技有限公司

原子哥在线教学平台：www.yuanzige.com

开源电子网 / 论坛 ：http://www.openedv.com/forum.php

正点原子淘宝店铺 ：https://openedv.taobao.com

正点原子官方网站 ：www.alientek.com

正点原子 B 站视频 ：

https://space.bilibili.com/394620890

电话：020-38271790 传真：020-36773971

请关注正点原子公众号，资料发布更新我们会通知。

请下载原子哥 APP，数千讲视频免费学习，更快更流畅。

扫码关注正点原子公众号

扫码下载“原子哥”APP

文档更新说明

版本	版本更新说明	负责人	校审	发布日期
V1.0	初稿：	正点原子 linux 团队	正点原子 linux 团队	2023.08.05

目录

前言

在正点原子的出厂系统里，默认是 Buildroot 作为出厂系统。里面有 OpenCV 库，可以通过 C++或者 python 编写相应的程序来使用 OpenCV。本文档是建立在 Buildroot 系统之上，Debian 系统，Yocto 不作说明。所有 Linux 文档如不明确使用系统版本，默认为 Buildroot 系统。

免责声明

为了得到最新版本的产品信息，请用户定时访问正点原子资料下载中心或者与淘宝正点原子旗舰店客服联系索取。感谢您的包容与支持。

第一章 OpenCV4 使用说明

正点原子 ATK-DLRK3568 开发板适合计算机视觉开发，四核 A55@2.0GHz 主频，足够运行日常大型应用。

本章我们介绍如何编译和运行各种情况的 OpenCV 程序。

1.1 OpenCV4 简介

OpenCV 旨在提供高效、跨平台的计算机视觉算法和工具，使开发者能够快速构建基于图像和视频的应用。它支持多种编程语言，如 C++、Python、Java 等，适用于 Windows、Linux、macOS、iOS 和 Android 等不同平台。

在正点原子的出厂系统（Buildroot），OpenCV4 的版本为 OpenCV 4.5.5。其中 OpenCV 库位于/usr/lib 下如下图。

```
root@ATK-DLRK356X:/usr/lib# ls libopencv_*
libopencv_calib3d.so            libopencv_imgcodecs.so
libopencv_calib3d.so.405        libopencv_imgcodecs.so.405
libopencv_calib3d.so.4.5.5      libopencv_imgcodecs.so.4.5.5
libopencv_core.so               libopencv_imgproc.so
libopencv_core.so.405           libopencv_imgproc.so.405
libopencv_core.so.4.5.5         libopencv_imgproc.so.4.5.5
libopencv_dnn.so                libopencv_ml.so
libopencv_dnn.so.405            libopencv_ml.so.405
libopencv_dnn.so.4.5.5          libopencv_ml.so.4.5.5
libopencv_features2d.so         libopencv_objdetect.so
libopencv_features2d.so.405     libopencv_objdetect.so.405
libopencv_features2d.so.4.5.5   libopencv_objdetect.so.4.5.5
libopencv_flann.so              libopencv_photo.so
libopencv_flann.so.405          libopencv_photo.so.405
libopencv_flann.so.4.5.5        libopencv_photo.so.4.5.5
libopencv_freetype.so           libopencv_stitching.so
libopencv_freetype.so.405       libopencv_stitching.so.405
libopencv_freetype.so.4.5.5     libopencv_stitching.so.4.5.5
libopencv_gapi.so               libopencv_videoio.so
libopencv_gapi.so.405           libopencv_videoio.so.405
libopencv_gapi.so.4.5.5         libopencv_videoio.so.4.5.5
libopencv_highgui.so            libopencv_video.so
libopencv_highgui.so.405        libopencv_video.so.405
libopencv_highgui.so.4.5.5      libopencv_video.so.4.5.5
root@ATK-DLRK356X:/usr/lib#
```

其中就包含 freetype（一个开源的字体渲染引擎），可以使用它来绘制文字，方便开者使用。同时也支持 Python 调用 OpenCV，接口路径如下。

```
root@ATK-DLRK356X:/usr/lib/python3.8/site-packages/cv2# ls
config-3.8.py  __init__.py          load_config_py3.py   python-3.8
config.py      __init__.pyc         load_config_py3.pyc  utils
config.pyc     load_config_py2.py   mat_wrapper
gapi           load_config_py2.pyc  misc
root@ATK-DLRK356X:/usr/lib/python3.8/site-packages/cv2#
```

好了介绍了库，现在我们就来介绍如何使用 OpenCV 吧。

1.2 一个简单的 OpenCV 应用程序

现在我们写一个简单的 OpenCV 应用程序。

1.2.1 C++版本 OpenCV 程序

C++版本：

编辑一个 opencv_test.cpp 文件，将以下代码写进这个文件中。

```
#include <opencv4/opencv2/opencv.hpp>

int main() {
    // 读取图片，这里读取开发板上的一个图片，可以替换自己的
    cv::Mat image = cv::imread("/opt/apps/src/logo/alientek_logo.png");

    // 检查是否成功读取图片
    if (image.empty()) {
        std::cout << "无法读取图片文件" << std::endl;
        return -1;
```

```
        }

        // 创建一个窗口来显示图片
        cv::namedWindow("Image", cv::WINDOW_AUTOSIZE);

        // 显示图片
        cv::imshow("Image", image);

        // 等待用户按下任意键，然后关闭窗口
        cv::waitKey(0);
        cv::destroyAllWindows();

        return 0;
    }
```

1.2.2 Python 版本 OpenCV 程序

编辑一个 opencv_test.py 文件，将以下代码写入到这个文件中，注意格式。

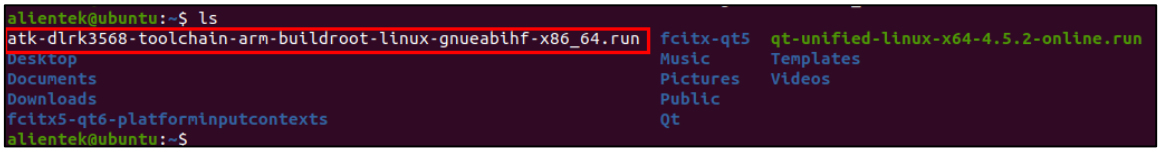
```
import cv2

def main():
    # 读取图片
    image = cv2.imread('/opt/apps/src/logo/alientek_logo.png')

    # 检查是否成功读取图片
    if image is None:
        print('无法读取图片文件')
        return

    # 创建一个窗口来显示图片
    cv2.namedWindow('Image', cv2.WINDOW_NORMAL)  # 使用 WINDOW_NORMAL 以允许调整窗口大小

    # 显示图片
    cv2.imshow('Image', image)

    # 等待用户按下任意键，然后关闭窗口
    cv2.waitKey(0)
    cv2.destroyAllWindows()

if __name__ == "__main__":
    main()
```

1.3 安装交叉编译工具链

注：如果你已经在其他文档里安装过交叉编译工具链，则不用重复安装。

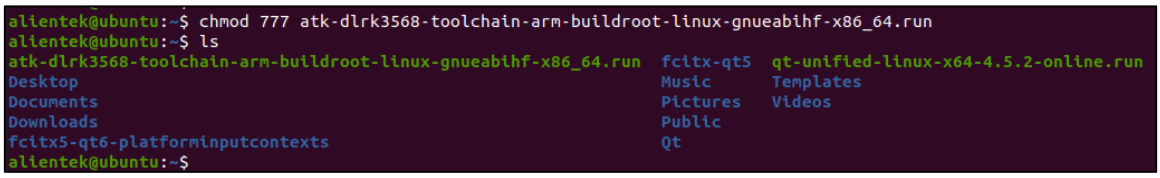
交叉编译工具链路径在我们 ATK-DLRK3568 光盘资料开发板光盘 A-基础资料->05、开发工具->01、交叉编译工具->atk-dlrk3568-toolchain-arm-buildroot-linux-gnueabihf-x86_64-(版本号).run。这个交叉编译工具是由正点原子在 RK3568 的 SDK 使用 Buildroot 中生成并打包。方便我们在不编译 ATK-DLRK3568 的 SDK 就能使用交叉编译工具。这也考虑到有某些开发者编译 SDK 有可能不通过，不想编译 SDK 就能方便的使用我们的交叉编译工具开发 Qt 应用程序。

将 atk-dlrk3568-toolchain-arm-buildroot-linux-gnueabihf-x86_64.run 这个文件拷贝到使用 WinScp 或者其它传输工具拷贝到 Ubuntu20.04 如下。

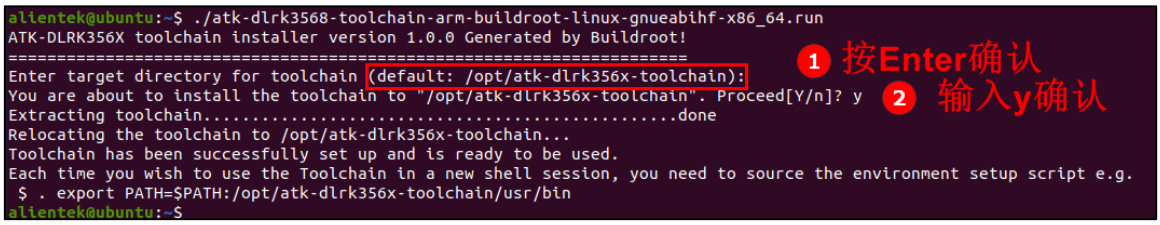


给 atk-dlrk3568-toolchain-arm-buildroot-linux-gnueabihf-x86_64.run 可执行权限。

```
chmod 777 atk-dlrk3568-toolchain-arm-buildroot-linux-gnueabihf-x86_64.run
```



执行下面指令安装。①步骤中可以指定交叉编译器的安装路径，默认是安装到/opt/atk-dlrk3568-toolchain 路径下。可以输入其他路径安装，但是这里我们不建议修改这个路径。②步骤中输入 y 或 Y 确认安装路径即可！保持交叉编译工具链安装路径与笔者的安装路径一样，下面步骤中也要用到。安装过程约 1~2 分钟，安装速度与个人计算机硬盘速度有关，安装完成如下。



1.4 C++版本 OpenCV4 程序

在 1.2 小节我们虽然写了一个简单的 OpenCV 应用 C++程序，但我们不知道如何编译它，然后拷贝到开发板上运行。所以我们总结了以下几种方法来编译你的 OpenCV 应用程序。

1.4.1 命令行版本

在 Ubuntu 创建一个 opencv 文件夹，创建 1.2.1 小节 opencv_test.cpp 文件。

执行下面指令，编译 opencv 程序。注意下面是一行程序，“\”的作用是自动换行。

```
/opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-g++ opencv_test.cpp -o opencv_test \
-I /opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/include/opencv4 \
-L/opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/lib \
-lopencv_core -lopencv_highgui -lopencv_imgproc -lopencv_videoio -lopencv_imgcodecs
```

执行结果如下图。

```
alientek@ubuntu:~/opencv$ /opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-g++ opencv_test.cpp -o opencv_test \
> -I /opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/include/opencv4 \
> -L/opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/lib \
> -lopencv_core -lopencv_highgui -lopencv_imgproc -lopencv_videoio -lopencv_imgcodecs
alientek@ubuntu:~/opencv$ ls
opencv_test  opencv_test.cpp
alientek@ubuntu:~/opencv$
```

指令解释：

/opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-g++： 使用 g++，如果是 c 程序使用 gcc。

opencv_test.cpp：程序源文件

-o opencv_test： -o 指输出文件，后面跟着是输出文件名

-I： 包括头文件路径。

-L：包括库文件路径。

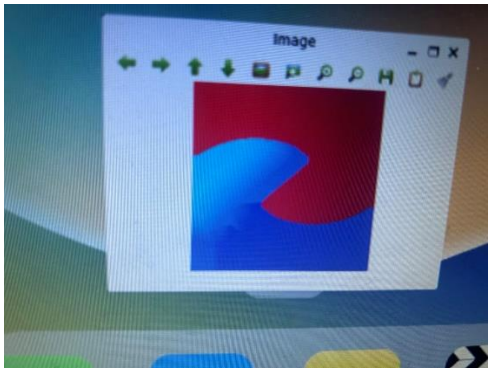
-l: 后面跟着的是库名字。比如笔者现在用到 opencv_core，也就是 opencv 核心模块，这种库名字可以在/opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/lib 下找，不过你得知道你用了哪种库，才需要链接到相应的库。

将此程序拷贝到开发板出厂系统上执行，默认出厂系统为 Buildroot 系统。拷贝的方法有很多种，可以参考 10、用户手册\03、辅助文档\14【正点原子】Ubuntu&Windows&Linux 开发板互传文件参考手册使用 scp 指令来拷贝，或者参考 10、用户手册\03、辅助文档\10【正点原子】adb 工具使用说明使用 adb 指令来传输这个 opencv_test 到开发板上执行。

如下图，开发板上执行如下。

```
root@ATK-DLRK356X:/# ./opencv_test
QStandardPaths: runtime directory '/var/run' is not a directory, but a symbolic link to a directory permissions 0755 owned by UID 0 GID 0
```

屏幕上显示的结果如下，显示一张正点原子的 logo，可以全屏放大。



1.4.2 Makefile 版本

Makefile 版本我们直接贴出代码，学习过 Makefile 都知道，下面代码就不再解释了。

在 Ubuntu 创建一个 Makefile，写入以下内容。

```
#@author Deng Zhimao
#@email dengzhimao@alientek.com
#http://www.openedv.com/forum.php
```

```
hide := @
ECHO := echo

TOOLCHAIN_DIR := /opt/atk-dlrk356x-toolchain
G++ := ${TOOLCHAIN_DIR}/bin/aarch64-buildroot-linux-gnu-g++
SYSROOT := ${TOOLCHAIN_DIR}/aarch64-buildroot-linux-gnu/sysroot/usr/include
CMAKE_SYSROOT := ${TOOLCHAIN_DIR}/aarch64-buildroot-linux-gnu/sysroot

CFLAGS += -I${SYSROOT} \
          -I${SYSROOT}/opencv4

CFLAGS += --sysroot=${CMAKE_SYSROOT}

LD_FLAGS := -lopencv_core -lopencv_videoio -lopencv_video \
            -lopencv_highgui -lopencv_imgcodecs -lopencv_imgproc

all:
        $(G++) opencv_test.cpp $(CFLAGS) $(LD_FLAGS) -o opencv_test
        $(hide)$(ECHO) "Build Done ..."

clean:
        rm opencv_test
```

完成如下图。

```
alientek@ubuntu:~/opencv$ ls
Makefile  opencv_test.cpp
alientek@ubuntu:~/opencv$
```

直接输入 make，就可以编译这个程序了，输入 make clean 则会清除编译的 opencv_test。同 1.4.1 小节一样拷贝生成的 opencv_test 到开发板上执行即可。

```
alientek@ubuntu:~/opencv$ make
/opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-g++ opencv_test.cpp
 -I/opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/include
-I/opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroot/usr/include/o
pencv4 --sysroot=/opt/atk-dlrk356x-toolchain/aarch64-buildroot-linux-gnu/sysroo
t -lopencv_core -lopencv_videoio -lopencv_video -lopencv_highgui -lopencv_imgco
decs -lopencv_imgproc -o opencv_test
Build Done ...
alientek@ubuntu:~/opencv$ ls
Makefile  opencv_test  opencv_test.cpp
alientek@ubuntu:~/opencv$ make clean
rm opencv_test
alientek@ubuntu:~/opencv$
```

1.4.3 CMakeLists 版本

CMake 提供了一种更抽象的配置方式，允许开发者使用简单的命令来指定项目的编译选项、依赖库和目标等。这相对于编写庞大复杂的 Makefile 更加直观和易于管理。

在源码的当前目录创建一个 CMakeLists.txt 文件，写入以下内容。内容也是很简单，设置工具路径和链接库而已。

```
#@author            Deng Zhimao
#@email             dengzhimao@alientek.com
#http://www.openedv.com/forum.php

cmake_minimum_required(VERSION 3.8)
message(STATUS "cmake version ${CMAKE_VERSION}")

set(TOOLCHAIN_DIR /opt/atk-dlrk356x-toolchain)
set(CMAKE_CXX_COMPILER ${TOOLCHAIN_DIR}/bin/aarch64-buildroot-linux-gnu-g++)
set(CMAKE_C_COMPILER ${TOOLCHAIN_DIR}/bin/aarch64-buildroot-linux-gnu-gcc)
set(SYSROOT ${TOOLCHAIN_DIR}/aarch64-buildroot-linux-gnu/sysroot/usr/include)
set(CMAKE_SYSROOT ${TOOLCHAIN_DIR}/aarch64-buildroot-linux-gnu/sysroot)

set(CMAKE_CXX_STANDARD 11)
set(OPENCV_LIBS opencv_core opencv_videoio opencv_video opencv_highgui
opencv_imgcodecs opencv_imgproc)

include_directories(${SYSROOT})
include_directories(${SYSROOT}/opencv4)

project(opencv)
#生成 opencv_test 执行程序
add_executable(opencv_test opencv_test.cpp)
target_link_libraries(opencv_test ${OPENCV_LIBS})
```

完成如下图。

```
alientek@ubuntu:~/opencv$ ls
CMakeLists.txt  Makefile  opencv_test.cpp
alientek@ubuntu:~/opencv$
```

上图在上一小节已经有了一个 Makefile，所以我们需要创建一个 build 目录，用于 CMake 生成新的 Makefile。

```
mkdir build     #创建一个 build 目录
cd build        #进入 build 目录
cmake ..        #cmake 根据上一级的 CMakeLists.txt 生成 Makefile 等文件
ls              #查看生成的文件
```

```
alientek@ubuntu:~/opencv$ mkdir build
alientek@ubuntu:~/opencv$ cd build/
alientek@ubuntu:~/opencv/build$ cmake ..
-- cmake version 3.16.3
-- The C compiler identification is GNU 10.3.0
-- The CXX compiler identification is GNU 10.3.0
-- Check for working C compiler: /opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-gcc
-- Check for working C compiler: /opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-g++
-- Check for working CXX compiler: /opt/atk-dlrk356x-toolchain/bin/aarch64-buildroot-linux-gnu-g++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/alientek/opencv/build
alientek@ubuntu:~/opencv/build$ ls
CMakeCache.txt  CMakeFiles  cmake_install.cmake  Makefile
alientek@ubuntu:~/opencv/build$
```

直接执行 make

```
make
```

```
alientek@ubuntu:~/opencv/build$ make
Scanning dependencies of target opencv_test
[ 50%] Building CXX object CMakeFiles/opencv_test.dir/opencv_test.cpp.o
[100%] Linking CXX executable opencv_test
[100%] Built target opencv_test
alientek@ubuntu:~/opencv/build$
```

同 1.4.1 小节一样拷贝生成的 opencv_test 到开发板上执行即可。

使用 ls 查看生成的二进制文件，如需要清除使用 make clean。

```
alientek@ubuntu:~/opencv/build$ ls
CMakeCache.txt  CMakeFiles  cmake_install.cmake  Makefile  opencv_test
alientek@ubuntu:~/opencv/build$ make clean
alientek@ubuntu:~/opencv/build$ ls
CMakeCache.txt  CMakeFiles  cmake_install.cmake  Makefile
alientek@ubuntu:~/opencv/build$
```

```
make clean
```

1.4.4 Qt 版本

在 Qt 项目里如何添加 OpenCV 进行编译呢？本小节以写了个模板，供大家参考。

注意：前提我们要配置好 ATK-DLRK3568 的交叉编译环境，请参考 10、用户手册\03、辅助文档\09【正点原子】ATK-DLRK3568_Qt 开发环境搭建的第一、二章。

笔者新建了一个 opencv_test Qt 项目测试。在 opencv_test.pro 里添加了以下内容。如下面代码红色部分。Qt 会根据 pkgconfig 的内容找到头文件和库。

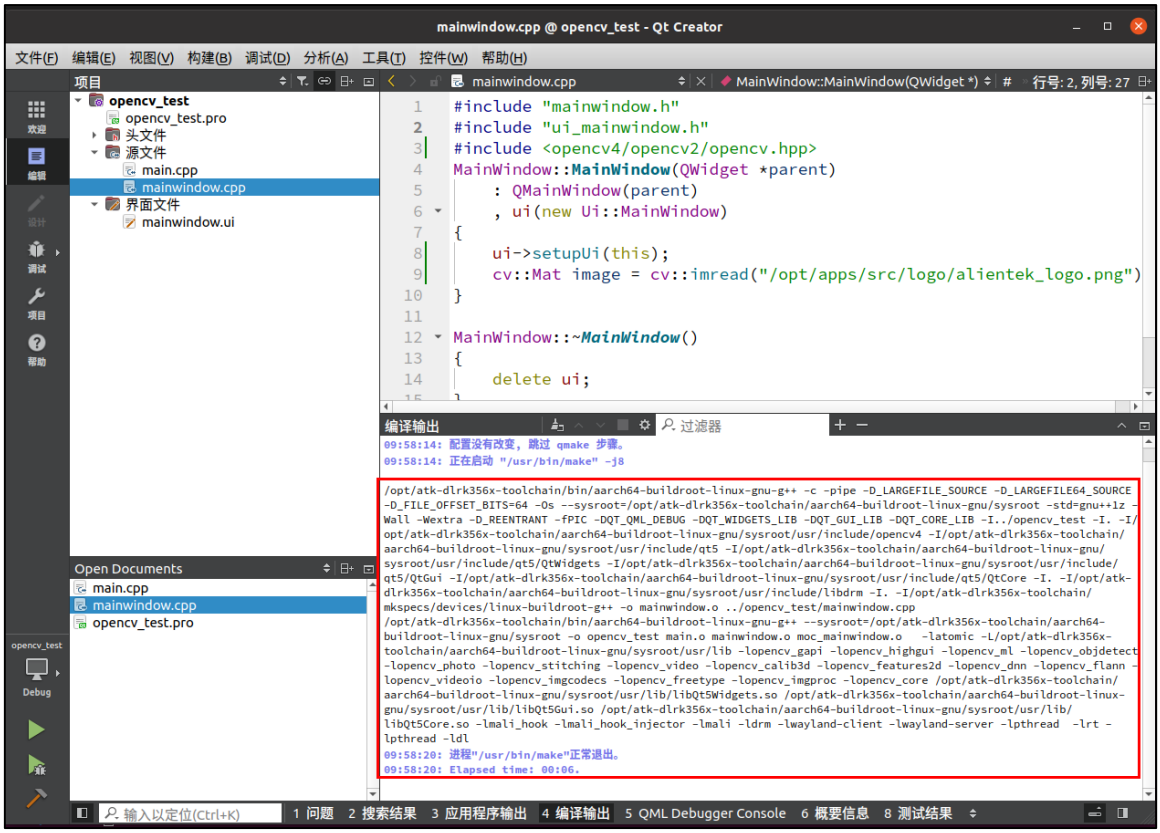
```
QT       += core gui

greaterThan(QT_MAJOR_VERSION, 4): QT += widgets

CONFIG += c++17

# You can make your code fail to compile if it uses deprecated APIs.
# In order to do so, uncomment the following line.
#DEFINES += QT_DISABLE_DEPRECATED_BEFORE=0x060000    # disables all the APIs deprecated before Qt 6.0.0

SOURCES += \
```

```
        main.cpp \
        mainwindow.cpp

HEADERS += \
        mainwindow.h

FORMS += \
        mainwindow.ui
CONFIG += link_pkgconfig
PKGCONFIG += opencv4

# Default rules for deployment.
qnx: target.path = /tmp/$${TARGET}/bin
else: unix:!android: target.path = /opt/$${TARGET}/bin
!isEmpty(target.path): INSTALLS += target
```

笔者在 manwindow.cpp 添加了以下红色部分代码测试。注以下代码并不实现具体功能，仅测试。

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include <opencv4/opencv2/opencv.hpp>
MainWindow::MainWindow(QWidget *parent)
        : QMainWindow(parent)
        , ui(new Ui::MainWindow)
{
        ui->setupUi(this);
        cv::Mat image = cv::imread("/opt/apps/src/logo/alientek_logo.png");
}

MainWindow::~MainWindow()
{
        delete ui;
}
```

开始构建，Ctrl+b 快捷键开始构建项目。可以看到下图，Qt 已经自动识别 opencv4 的路径并且能链接到所有库，而不用手动指定。构建成功如下图。



1.5 Python 版本 OpenCV4 程序

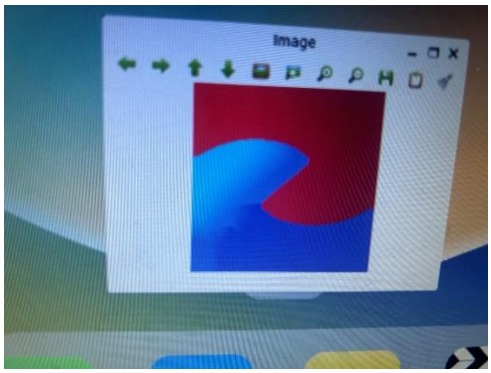
python 版本的 Opencv 程序则简单许多，因为 python 程序不需要编译，由 python 解释器解释。将 1.2.2 的 pyhton 程序拷贝到开发板上直接运行，如下图。

```
python3 opencv_test.py
```

```
root@ATK-DLRK356X:/# python3 opencv_test.py
QStandardPaths: runtime directory '/var/run' is not a directory, but a symbolic link to a directory permissions 0755 owned by UID 0 GID 0
```

程序执行的结果与前面 C++版本的一样，如下图。

屏幕上显示的结果如下，显示一张正点原子的 logo，可以全屏放大。



附录-常见问题

Q：如果没有 MIPI/LVDS 屏幕，使用 HDMI 显示器，Weston 桌面只显示在显示中间，如何做？

A：请参考 08、RK 官方文档\01、Linux\Linux\Graphics\Rockchip_Developer_Guide_Buildroot_Weston_CN.pdf。将 HDMI 旋转 90 度或者将 HDMI 显示设置为主屏（主显示）。