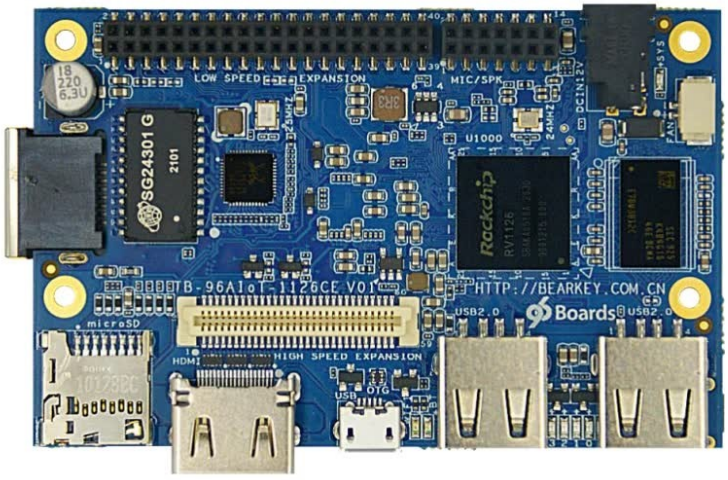
# RV1126 CE Boards

# Software User Guide



# Content

# 1 Buildroot & Debian on the RV1126 CE Boards

The RV1126 CE Boards software support two os systems: Buildroot and Debian. Debian image for the RV1126 CE Boards are built is based on Debian 9 version.

# 2. Installing Buildroot & Debian

The currently supported method to install a Buildroot or Debian Image on the RV1126 CE Boards:

- Installing the image from a Host computer via a USB cable and Rockchip Flash Download Tool

The following chapters describe the method in detail.

## 2.1 Installing the image from a Host-pc

This method is recommended for experienced users who will be downloading many iterative experimental versions of self-compiled OS's. It is also a fallback method in case the first method fails because either the monitor or mouse and keyboard could not be detected. Since this method uses a HostPC to program the board a separate monitor and mouse/keyboard do not need to be connected to the board. This guide describes the process Windows Host systems.

### 2.1.1 Installation prerequisites

This method requires the rockchip flash download tool and corresponding driver to be installed on the HostPC. rockchip flash download tool is a tool that communicates with the bootloader of the RV1126 CE Boards and allows you to flash images onto the board. See below for instruction on how to install rockchip flash download tool dirver on your Host PC.

### 2.1.2 Installation overview

In order to install Buildroot from a Host PC just follow these simple steps:

step 1. Download the Buildroot & Debian images

step 2. Install Rockchip Flash Download Driver in Host machine

step 3. Bring the board into maskrom or rockusb loader mode

step 4. Start the rockchip flash download tool on the HostPC

step 5. Flash the buildroot image
step 6. Reboot and enjoy

The following chapters describe each step in more detail:

### 2.1.3 Step1. Download the Buildroot & Debian images and flash download tool

1. Download the buildroot & debian images from below network disk:
https://1drv.ms/u/s!AhoFWWcHV7rXklDzYPeAll4WLSRx?e=Ff0usx

2. Download rockchip flash download tools( window & linux version) from below network disk:
https://1drv.ms/u/s!AhoFWWcHV7rXklDzYPeAll4WLSRx?e=Ff0usx

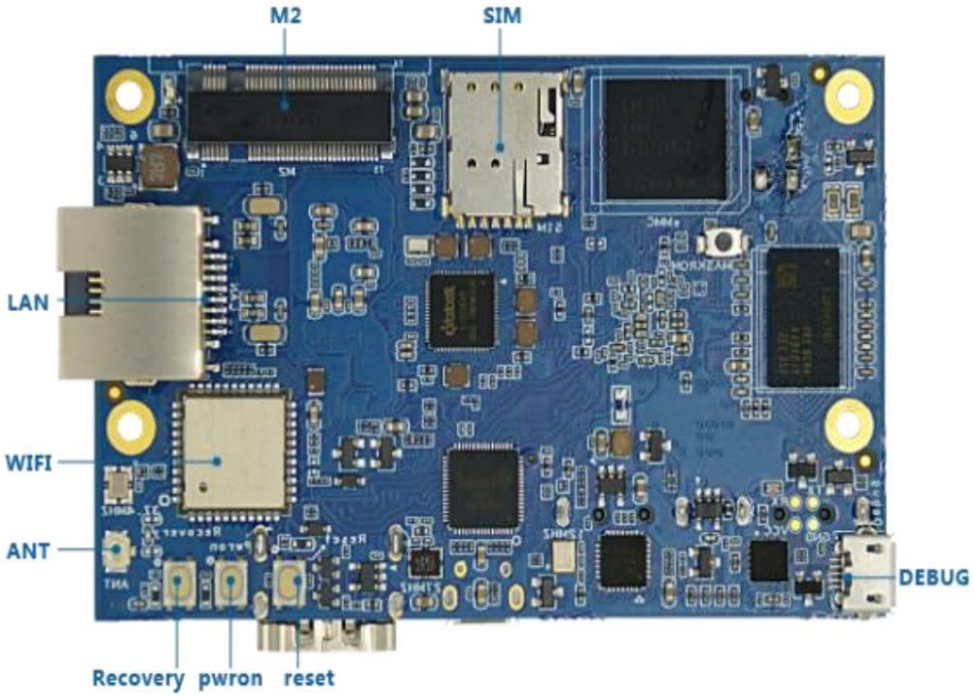
### 2.1.4 step 2. Install Rockchip Flash Download Driver in Host machine

In the network disk tools Directory: tools\windows\DriverAssitant_v5.0.zip

Double click DriverAssitant_v5.0\ DriverInstall.exe，then automatically install rockchip flash download driver in your host windows machine.

### 2.1.5 step 3. Bring the board into maskrom or rockusb loader mode

Use typec usb cable to connect RV1126 CE Boards and Host PC. Then put your RV1126 CE Boards enter download mode(maskrom or rockusb loader mode). The method of download mode as follows: Firstly press reset key of the developing board, and then long press recovery key around 3-4 seconds to enter.



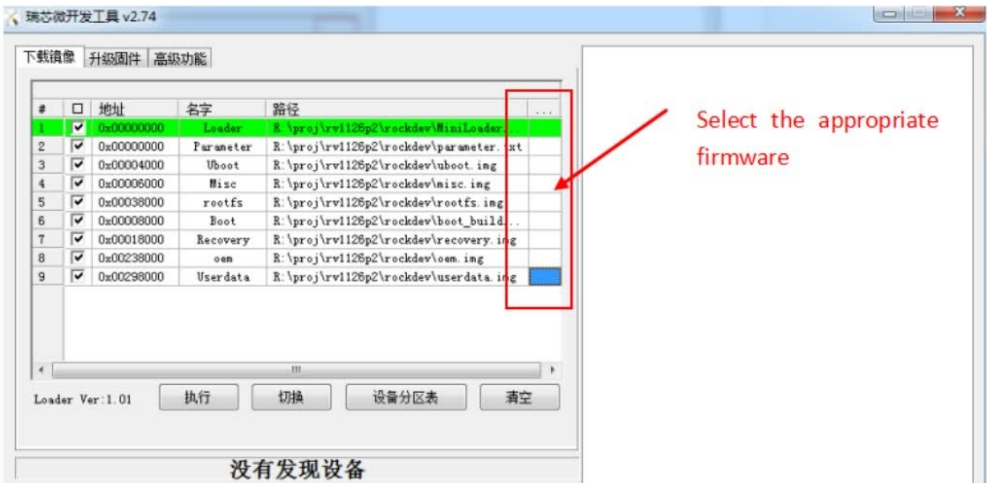

## 2.1.6 step 4. Start the rockchip flash download tool on the HostPC

**Host PC OS is Windows:**

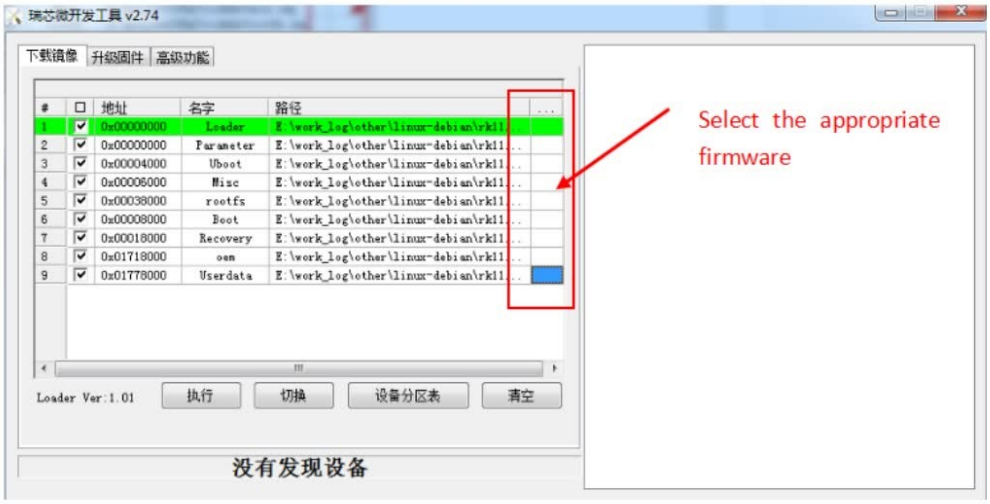
In the network disk tools Directory: tools\windows\RKDevTool_Release.zip

Double click RKDevTool_Release\RKDevTool.exe.

Flash buildroot system images:



Flash debian system images:




**Host PC OS is Linux:**

In the network disk tools Directory: tools\linux\Linux_Upgrade_Tool_v1.57.zip

Unzip Linux_Upgrade_Tool_v1.57.zip and unzip the images into Linux_Upgrade_Tool folder.Such as: rv1126-buildroot, rv1126-debian.

## 2.1.7 step 5. Flash the buildroot & debian images

**Host PC OS is Windows:**

Open the tool, and click “Download Image” menu. Single click every line end as marked with red arrow, it will pop out file selection box and then choose the img file path of the corresponding partition.Set all the img file paths successively.After configuration, click “Run”. The right information box will display the relative information.

**Host PC OS is Linux:**

1. Flash buildroot images

```
sudo ./upgrade_tool ul rv1126-buildroot/MiniLoaderAll.bin
sudo ./upgrade_tool di -p rv1126-buildroot/parameter.txt
sudo ./upgrade_tool di -u rv1126-buildroot/uboot.img
sudo ./upgrade_tool di -misc rv1126-buildroot/misc.img
sudo ./upgrade_tool di -b rv1126-buildroot/boot.img
sudo ./upgrade_tool di -recovery rv1126-buildroot/recovery.img
sudo ./upgrade_tool di -oem rv1126-buildroot/oem.img
sudo ./upgrade_tool di -rootfs rv1126-buildroot/rootfs.img
sudo ./upgrade_tool di -userdata rv1126-buildroot/userdata.img
```

```
sudo ./upgrade_tool rd

2. Flash debian images
sudo ./upgrade_tool ul rv1126-debian/MiniLoaderAll.bin
sudo ./upgrade_tool di -p rv1126-debian/parameter.txt
sudo ./upgrade_tool di -u rv1126-debian/uboot.img
sudo ./upgrade_tool di -misc rv1126-debian/misc.img
sudo ./upgrade_tool di -b rv1126-debian/boot.img
sudo ./upgrade_tool di -recovery rv1126-debian/recovery.img
sudo ./upgrade_tool di -oem rv1126-debian/oem.img
sudo ./upgrade_tool di -rootfs rv1126-debian/rootfs.img
sudo ./upgrade_tool di -userdata rv1126-debian/userdata.img
sudo ./upgrade_tool rd
```

### 2.1.8 Step 6. Reboot and enjoy!

Once the download of the images is complete, follow these steps:

- Disconnect the USB cable
- Press the reset button

If you flash buildroot images，after the reboot you should see buildroot startup.
If you flash debian images，after the reboot you should see debian startup.

# 3. Building Buildroot Source

This section provides the instructions for building and Buildroot system on RV1126 CE Boards from x86 host machine.

## 3.1 Download GNU cross-toolchain binaries and Installing required packages(Host machine Ubuntu 14.04 or Ubuntu 16.04)

You need to download the correct GCC toolchain depending your host/target architecture. Usually host is a standard Intel x86-64 computer by Ubuntu 16.04 or 18.04 , target is the RV1126 CE Boards which is Arm ARMHF.

```
$ sudo apt-get install repo device-tree-compiler git-core u-bo
ot-tools mtools parted libudev-dev libusb-1.0-0-dev python-lin
```

```
aro-image-tools linaro-image-tools autoconf autotools-dev libs
igsegv2 m4 intltool libdrm-dev curl sed make binutils build-es
sential gcc g++ bash patch gzip gawk bzip2 perl tar cpio pytho
n unzip rsync file bc wget libncurses5 libqt4-dev libglib2.0-d
ev libgtk2.0-dev libglade2-dev cvs git mercurial openssh-clien
t subversion asciidoc w3m dblatex graphviz python-matplotlib l
ibc6:i386 libssl-dev expect fakeroot cmake flex bison liblz4-t
ool libtool keychain
```

## 3.2 Get RV1126 CE Boards Buildroot source code

```
# repo init --repo-url=https://gitlab.com/bearkey-tools/repo -u \
  https://gitlab.com/bearkey-1126-linux/manifests.git --no-clone-bundle
```

```
# repo sync
```

## 3.3 Select Board config

```
$ ./build.sh BoardConfig-rv1126-bearkey-ce.mk
```

## 3.4 Build Uboot

```
$ ./build.sh uboot
```

## 3.5 Build Linux kernel

```
$ ./build.sh kernel
```

## 3.6 Build Rootfs

```
$ ./build.sh rootfs
```

## 3.7 Build Recovery

```
$ ./build.sh recovery
```

## 3.8 Build all

```
$ ./build.sh
```

## 3.9 Generate RV1126 CE Boards Buildroot image for flash download

```
$ ./mkfirmware.sh
```

Ps:The generated firmware is in the rockdev directory of the SDK.

# 4. Develop Buildroot Driver Sample

## 4.1 Simple Driver Sample

### 4.1.1 Enter kernel driver patch and create hello dir and hello.c:

```
$ cd kernel/drivers

$ mkdir hello

$ cd hello

$ vim hello.c
```

### 4.1.2 Start coding in hello.c:

```
#include <linux/init.h>

#include <linux/module.h>

#include <linux/kernel.h>



MODULE_LICENSE("Dual BSD/GPL");

static int hello_init(void)  {

         printk(KERN_ALERT "hello,I am a simple driver sample\n
");

        return 0;

}

static void hello_exit(void)  {

        printk(KERN_ALERT "goodbye,kernel\n");

}

module_init(hello_init);

module_exit(hello_exit);

MODULE_AUTHOR("Beiqi");

MODULE_DESCRIPTION("This is a simple driver example!\n");

MODULE_ALIAS("A simplest example");
```

### 4.1.3 Create Makefile and Kconfig:

```
//Makefile
```

```
 obj-$(CONFIG_BEIQI_HELLO)              += hello.o

//Kconfig

#

#gpio device configuration

#

menuconfig BEIQI_HELLO

       bool "support beiqi hello test devices"

       default y

       ---help---

        Say Y here if you support beiqi hello device
```

### 4.1.4 Modify Makefile:

```
$ cd ..

$ git diff

diff --git a/drivers/Makefile b/drivers/Makefile

index 4fa22d2..62a54d1 100644

--- a/drivers/Makefile

+++ b/drivers/Makefile

@@ -192,3 +192,4 @@ obj-$(CONFIG_RK_NAND)              += rk_nand/

 obj-$(CONFIG_RK_HEADSET)       += headset_observe/

 obj-$(CONFIG_RK_FLASH)         += rkflash/

 obj-$(CONFIG_BEIQI_ADC)         += beiqi_adc/
```

```
+obj-m$(CONFIG_BEIQI_HELLO)             += hello/hello.o
```

### 4.1.5 Build module:

```
$ cd ../../

$ ./build.sh kernel

$ ls kernel/drivers/hello/

hello.c  hello.ko  hello.mod.c  hello.mod.o  hello.o  Kconfig  Mak
efile
```

### 4.1.6 install module:

```
$ sudo insmod hello.ko
```

### 4.1.7 run to see log:

```
$ dmesg | tail -5
```

### 4.1.8 unstall module:

```
$ sudo rmmod hello.ko
```

# 5. Develop Buildroot Application Sample

Builderroot add compiler application, please refer to the following.

## 5.1 Simple Buildroot Application Sample

### 5.1.1 Source code location

Select the location of the source code in the SDK, for example:external.Take adding a hello app as an example.

```
$ ls external/hello

hello.c  Makefile

$ cat external/hello/hello.c

#include <stdio.h>

int main(void)

{

    printf("hello world,I am a simple linux application sample\n
");

    return 0;

}

$ cat external/hello/Makefile

CROSS_COMPILE=arm-linux-gnueabi-

CC = ${CROSS_COMPILE}gcc

CXX = ${CROSS_COMPILE}g++

LIBS += -lm

LIBS += -ldl

CFLAGS += -I./include -I$(STAGING_DIR)/usr/include

CFLAGS += -O2

#CFLAGS += -fno-stack-protector

LDFLAGS += -L$(STAGING_DIR)/usr/lib -L./
```

```
OBJECTS = $(SOURCES:.c=.o)

PRJOBJS = hello.c

PRJNAME = hello

all: ${PRJNAME}

$(PRJNAME):$(OBJECTS) $(PRJOBJS)

	$(CC) -o $@ $(OBJECTS) $(PRJOBJS) $(LIBS) $(LDFLAGS) $(CFLA
GS)

.c.o:

	$(CC)  $(INC) $(CFLAGS) -c -o $@ $<

.cpp.o:

	$(CXX)  $(INC) $(CFLAGS) -c -o $@ $<

clean:

	rm -rf $(PRJNAME) *.so *.o
```

### 5.1.2 Create configuration in builderroot

Select the package configuration location for builderoot, for example:buildroot/ package/rockchip.Take adding a hello app as an example.

```
$ ls buildroot/package/rockchip/hello

Config.in  hello.mk

$ cat buildroot/package/rockchip/hello/Config.in

config BR2_PACKAGE_HELLO

	bool "hello"

	help

		All files for hello
```

```
$ cat buildroot/package/rockchip/hello/hello.mk

HELLO_SITE_METHOD = local

HELLO_SITE = ${TOPDIR}/../external/hello

HELLO_DEPENDENCIES =

define HELLO_BUILD_CMDS

	$(TARGET_MAKE_ENV) $(MAKE) CC=$(TARGET_CC) CXX=$(TARGET_CX
X) -C $(@D)

endef

define HELLO_INSTALL_TARGET_CMDS

	cp $(@D)/hello ${TARGET_DIR}/usr/bin

endef

$(eval $(generic-package))

	$(CXX)  $(INC) $(CFLAGS) -c -o $@ $<
```

### 5.1.3 Add Hello application to buildroot menu

Add the following:

```
$ git diff buildroot/package/rockchip/Config.in

+source "package/rockchip/hello/Config.in"
```

### 5.1.4 Join a project configuration

Select the rootfs configuration for project compilation.In boardconfig-ce.mk You can see the configuration of rootfs is rockchip_rv1126_rv1109_defconfig.

```
$ git diff buildroot/configs/rockchip_rv1126_rv1109_defconfig

+BR2_PACKAGE_HELLO=y
```

### 5.1.5 Compiling Hello application

```
$ ./buildroot/build/envsetup.sh

This script is executed directly...

Top of tree: /home/qianchen/proj/rv1126p2



You're building on Linux

Lunch menu...pick a combo:



0. non-rockchip boards

1. rockchip_px30_32

2. rockchip_px30_64

78. rockchip_rv1126_rv1109

79. rockchip_rv1126_rv1109_ab

80. rockchip_rv1126_rv1109_facial_gate

81. rockchip_rv1126_rv1109_libs

82. rockchip_rv1126_rv1109_ramboot_uvcc

83. rockchip_rv1126_rv1109_recovery

84. rockchip_rv1126_rv1109_sl

85. rockchip_rv1126_rv1109_spi_nand

86. rockchip_rv1126_rv1109_spi_nand_recovery

87. rockchip_rv1126_rv1109_systemd
```

```
88. rockchip_rv1126_rv1109_tiny

89. rockchip_rv1126_rv1109_toolchain

90. rockchip_rv1126_rv1109_uvcc

91. rockchip_rv1126_rv1109_uvcc_spi_nand

Which would you like? [0]: 78

$ make hello  //compile

$ make hello-dirclean  //clean

 $ ls buildroot/output/rockchip_rv1126_rv1109/build/hello  //demo
hello
```

The above is just an example. You need to understand the builderroot compiler to understand the official documents.

# 6. Develop Linux(Debian) Application Sample

Debian is a linux desktop system. Develop debian application is the same as linux. So we develop linux application sample as follows.

## 6.1 Simple Linux Application Sample

### 6.1.1 create helloworld dir and helloworld.c:

```
$ mkdir helloworld

$ vim helloworld.c
```

### 6.1.2 Start coding in helloworld.c:

```
#include <stdio.h>
```

```
int main(void)

{

        printf("hello world,I am a simple linux application sample\
n");

        return 0;

}
```

### 6.1.3 Build application:

```
$ gcc helloworld.c -o helloworld
```

### 6.1.4 run application and see output:

```
$ ./helloworld
```

**run output is :**

hello world,I am a simple linux application sample

## 6.2 Simple rga demo

### 6.2.1 rga demo source code and analysis:

```
//rga_test.c

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <png.h>

#include <rockchip/RgaApi.h>
```

```
#include <rockchip/RgaApi.h>
```



```
#define PNG_BYTES_TO_CHECK    8

#define HAVE_ALPHA           0

#define NOT_HAVE_ALPHA        1


typedef struct _pic_data pic_data;

struct _pic_data

{

    int width, height;

    int bit_depth;

    int alpha_flag;

    unsigned char *rgba;

};


int check_is_png(FILE **fp, const char *filename)

{

    char checkheader[PNG_BYTES_TO_CHECK];

    *fp = fopen(filename, "rb");

    if (*fp == NULL) {

        printf("open failed ...1\n");

        return -1;

    }
```

```
    if (fread(checkheader, 1, PNG_BYTES_TO_CHECK, *fp) != PNG_BYTE
S_TO_CHECK) //Read PNG file length error, exit directly

        return 0;

    return png_sig_cmp(checkheader, 0, PNG_BYTES_TO_CHECK);

}

int decode_png(const char *filename, pic_data *out)

{

    png_structp png_ptr;

    png_infop    info_ptr;

    int ret;

    FILE *fp;

    if (check_is_png(&fp, filename) != 0) {

        printf("file is not png ...\n");

        return -1;

    }

    printf("launcher[%s] ...\n", PNG_LIBPNG_VER_STRING); //Print t
he current libpng version number

    //1: Initialize the data structure of libpng :png_ptr, info_pt
r

    png_ptr  = png_create_read_struct(PNG_LIBPNG_VER_STRING, NULL,
 NULL, NULL);

    info_ptr = png_create_info_struct(png_ptr);

    //2: Set the return point of the error

    setjmp(png_jmpbuf(png_ptr));
```

```
    rewind(fp); //or fseek(fp, 0, SEEK_SET);

    //3: Bind PNG structure to file stream IO

    png_init_io(png_ptr, fp);

    //4:Read PNG file information and convert to RGBA:8888 data fo
rmat

    png_read_png(png_ptr, info_ptr, PNG_TRANSFORM_EXPAND, 0); //Re
ad file information

    int channels, color_type;

    channels     = png_get_channels(png_ptr, info_ptr);

    color_type     = png_get_color_type(png_ptr, info_ptr);

    out->bit_depth = png_get_bit_depth(png_ptr, info_ptr);

    out->width      = png_get_image_width(png_ptr, info_ptr);

    out->height  = png_get_image_height(png_ptr, info_ptr);

    printf("channels = %d color_type = %d bit_depth = %d width = %d
 height = %d ...\n",

           channels, color_type, out->bit_depth, out->width, out->
height);



    int i, j, k;

    int size, pos = 0;

    int temp;

    //5: Read the actual RGB data

    png_bytepp row_pointers; //buf for storing RGB data

    row_pointers = png_get_rows(png_ptr, info_ptr);
```

```
    size = out->width * out->height; //Calculate space before appl
ying for memory

    if (channels == 4 || color_type == PNG_COLOR_TYPE_RGB_ALPHA)
{ //Judge whether it is 24 bits or 32 bits

        out->alpha_flag = HAVE_ALPHA; //Record whether there is a t
ransparent channel

        size *= (sizeof(unsigned char) * 4); //size = out->width *
out->height * channel

        out->rgba = (png_bytep)malloc(size);

        if (NULL == out->rgba) {

            printf("malloc rgba faile ...\n");

            png_destroy_read_struct(&png_ptr, &info_ptr, 0);

            fclose(fp);

            return -1;

        }

        //from row_pointers out the actual RGB data in the pointers

        temp = channels - 1;

        for (i = 0; i < out->height; i++)

            for (j = 0; j < out->width * 4; j += 4)

                for (k = temp; k >= 0; k--)

                    out->rgba[pos++] = row_pointers[i][j + k];

    } else if (channels == 3 || color_type == PNG_COLOR_TYPE_RGB)
{

        out->alpha_flag = NOT_HAVE_ALPHA;
```

```
        size *= (sizeof(unsigned char) * 3);
```



```
        out->rgba = (png_bytep)malloc(size);
        if (NULL == out->rgba) {
            printf("malloc rgba faile ...\n");
            png_destroy_read_struct(&png_ptr, &info_ptr, 0);
            fclose(fp);
            return -1;
        }
        temp = (3 * out->width);
        for (i = 0; i < out->height; i ++) {
            for (j = 0; j < temp; j += 3) {
                out->rgba[pos++] = row_pointers[i][j+2];
                out->rgba[pos++] = row_pointers[i][j+1];
                out->rgba[pos++] = row_pointers[i][j+0];
            }
        }
    } else return -1;
    //6:Destroy memory
    png_destroy_read_struct(&png_ptr, &info_ptr, 0);
    fclose(fp);
    //At this point, our out->rgba already stores the actual RGB d
ata
    //After processing free(out->rgba)
```

```
    return 0;
```



```
}

 int write_png_file(const char *filename , pic_data *out)

{

    png_structp png_ptr;

    png_infop     info_ptr;

    png_byte color_type;

    png_bytep * row_pointers;

    FILE *fp = fopen(filename, "wb");

    if (NULL == fp) {

        printf("open failed ...2\n");

        return -1;

    }

    //1: Initialize libpng structure

    png_ptr    = png_create_write_struct(PNG_LIBPNG_VER_STRING, 0,
0, 0);

    if (!png_ptr) {

        printf("png_create_write_struct failed ...\n");

        return -1;

    }

    //2: Initialize png_ Infop structure ,

    //this structure contains various information about the image,
such as size, pixel bit depth, color type, etc

    info_ptr = png_create_info_struct(png_ptr);
```

```
    if (!info_ptr) {
        printf("png_create_info_struct failed ...\n");
        return -1;
    }
    //3: Set the return point of the error
    if (setjmp(png_jmpbuf(png_ptr))) {
        printf("error during init_io ...\n");
        return -1;
    }
    //4:Bind PNG structure to file stream IO
    png_init_io(png_ptr, fp);
    if (setjmp(png_jmpbuf(png_ptr))) {
        printf("error during init_io ...\n");
        return -1;
    }
    if (out->alpha_flag == HAVE_ALPHA) color_type = PNG_COLOR_TYPE
_RGB_ALPHA;
    else color_type = PNG_COLOR_TYPE_RGB;
    //5: Setting and writing header information to PNG file
    png_set_IHDR(png_ptr, info_ptr, out->width, out->height, out->
bit_depth,
    color_type, PNG_INTERLACE_NONE, PNG_COMPRESSION_TYPE_BASE, PNG
_FILTER_TYPE_BASE);
```

```
    png_write_info(png_ptr, info_ptr);
```



```
    if (setjmp(png_jmpbuf(png_ptr))) {
        printf("error during init_io ...\n");
        return -1;
    }
    int channels, temp;
    int i, j, pos = 0;
    if (out->alpha_flag == HAVE_ALPHA) {
        channels = 4;
        temp = (4 * out->width);
        printf("have alpha ...\n");
    } else {
        channels = 3;
        temp = (3 * out->width);
        printf("not have alpha ...\n");
    }
    row_pointers = (png_bytep*)malloc(out->height * sizeof(png_byt
ep));
    for (i = 0; i < out->height; i++) {
        row_pointers[i] = (png_bytep)malloc(temp* sizeof(unsigned
char));
        for (j = 0; j < temp; j += channels) {
            if (channels == 4) {
                row_pointers[i][j+3] = out->rgba[pos++];
```

```
            row_pointers[i][j+2] = out->rgba[pos++];
            row_pointers[i][j+1] = out->rgba[pos++];
            row_pointers[i][j+0] = out->rgba[pos++];
        } else {
            row_pointers[i][j+2] = out->rgba[pos++];
            row_pointers[i][j+1] = out->rgba[pos++];
            row_pointers[i][j+0] = out->rgba[pos++];
        }
    }
}
//6: Write RGB data to PNG file
png_write_image(png_ptr, (png_bytepp)row_pointers);
if (setjmp(png_jmpbuf(png_ptr))) {
    printf("error during init_io ...\n");
    return -1;
}
//7: Write tail information
png_write_end(png_ptr, NULL);
//8:Free memory and destroy PNG structure
for (i = 0; i < out->height; i ++)
    free(row_pointers[i]);
free(row_pointers);
```

```
    png_destroy_write_struct(&png_ptr, &info_ptr);

    fclose(fp);

    return 0;

}

int main()

{

    pic_data out;

    char pi[] = "./m1.png";

    char po[] = "./m2.png";

    bo_t bo_src, bo_dst;

    int sf = RK_FORMAT_RGB_888;

    int df = RK_FORMAT_RGB_888;

    int sw = 240;

    int sh = 260;

    int dw = 240;

    int dh = 260;

    int bpp = 3;

    int ret;

    ////////////////////init rga//////////////////////////

    ret = c_RkRgaInit();

    ret = c_RkRgaGetAllocBuffer(&bo_src, sw, sh, 24);

    ret = c_RkRgaGetAllocBuffer(&bo_dst, dw, dh, 24);
```

```
    ret = c_RkRgaGetMmap(&bo_src);
    ret = c_RkRgaGetMmap(&bo_dst);
    ////////////////////get src data//////////////////////////
    decode_png(pi, &out);
//    memcpy(bo_src.ptr, out.rgba, sw * sh * 3);
    ////////////////////start rga//////////////////////////
    rga_info_t src;
    rga_info_t dst;
    memset(&src, 0, sizeof(rga_info_t));
    src.fd = -1;
    src.virAddr = out.rgba;
    src.mmuFlag = 1;
    memset(&dst, 0, sizeof(rga_info_t));
    dst.fd = -1;
    dst.virAddr = bo_src.ptr;
    dst.mmuFlag = 1;
    rga_set_rect(&src.rect, 0, 0, sw, sh, sw, sh, sf);
    rga_set_rect(&dst.rect, 0, 0, dw, dh, dw, dh, df);
    src.rotation = HAL_TRANSFORM_ROT_180;
    //src.rotation = HAL_TRANSFORM_ROT_90;
    ret = c_RkRgaBlit(&src, &dst, NULL);
    ////////////////////write data//////////////////////////
```

```
    memcpy(out.rgba, bo_src.ptr, sw * sh * 3);

    write_png_file(po, &out);

    free(out.rgba);

    return 0;

}
```

### 6.2.2 compile demo:

```
$ gcc rga_test.c -o rga_test -ldrm -lrga -lpng
```

### 6.2.3 run demo and see output:

```
$ ./rga_test

librga:RGA_GET_VERSION:3.02,3.020000

ctx=0x48b058,ctx->rgaFd=3

Rga built version:01e4e4b

launcher[1.6.28] ...

channels = 3 color_type = 2 bit_depth = 8 width = 240 height = 260
 ...

not have alpha ...

$ ls

m1.png  m2.png  rga_test
```

**ps :**

Put an image of m1.png in the current test directory.It will generate m2.png.

## 6.3 Simple mpp demo

## 6.1.4 mpp demo source code and analysis:

```
//dec_mpp_test.c

#define MODULE_TAG "mpi_dec_test"

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

#include "rockchip/rk_mpi.h"

#include "rockchip/mpp_buffer.h"

#include "rockchip/mpp_frame.h"



#define msleep(x)             usleep((x)*1000)

#define SZ_1K                 (1024)

#define SZ_4K                 (SZ_1K*4)

#define MPI_DEC_STREAM_SIZE       (SZ_4K)

#define MPI_DEC_LOOP_COUNT        4

#define MAX_FILE_NAME_LENGTH      256

#define mpp_malloc(type, count)  \

       (type*)malloc(sizeof(type) * (count))

#define mpp_calloc(type, count)  \

       (type*)malloc(sizeof(type) * (count))
```

```
#define mpp_free(ptr) \
        free(ptr)


typedef struct
{
    MppCtx          ctx;
    MppApi          *mpi;
    RK_U32          eos;
    char            *buf;
    MppBufferGroup  frm_grp;
    MppBufferGroup  pkt_grp;
    MppPacket       packet;
    size_t          packet_size;
    MppFrame        frame;
    FILE            *fp_input;
    FILE            *fp_output;
    RK_S32          frame_count;
    RK_S32          frame_num;
    size_t          max_usage;
} MpiDecLoopData;
typedef struct
{
```

```
    char            file_input[MAX_FILE_NAME_LENGTH];
    char            file_output[MAX_FILE_NAME_LENGTH];
    MppCodingType   type;
    MppFrameFormat  format;
    RK_U32          width;
    RK_U32          height;
    RK_U32          debug;
    RK_U32          have_input;
    RK_U32          have_output;
    RK_U32          simple;
    RK_S32          timeout;
    RK_S32          frame_num;
    size_t          max_usage;
} MpiDecTestCmd;

void dump_mpp_frame_to_file(MppFrame frame, FILE *fp)
{
    RK_U32 width    = 0;
    RK_U32 height   = 0;
    RK_U32 h_stride = 0;
    RK_U32 v_stride = 0;
    MppBuffer buffer    = NULL;
    RK_U8 *base = NULL;
```

```
width    = mpp_frame_get_width(frame);
height   = mpp_frame_get_height(frame);
h_stride = mpp_frame_get_hor_stride(frame);
v_stride = mpp_frame_get_ver_stride(frame);
buffer   = mpp_frame_get_buffer(frame);
base = (RK_U8 *)mpp_buffer_get_ptr(buffer);
RK_U32 buf_size = mpp_frame_get_buf_size(frame);
size_t base_length = mpp_buffer_get_size(buffer);
printf("base_length = %d\n",base_length);


RK_U32 i;
RK_U8 *base_y = base;
RK_U8 *base_c = base + h_stride * v_stride;
//Save to yuv420sp format
/*for (i = 0; i < height; i++, base_y += h_stride)
{
    fwrite(base_y, 1, width, fp);
}
for (i = 0; i < height / 2; i++, base_c += h_stride)
{
    fwrite(base_c, 1, width, fp);
}*/
```

```
    //Save to yuv420p format
    for(i = 0; i < height; i++, base_y += h_stride)
    {
        fwrite(base_y, 1, width, fp);
    }
    for(i = 0; i < height * width / 2; i+=2)
    {
        fwrite((base_c + i), 1, 1, fp);
    }
    for(i = 1; i < height * width / 2; i+=2)
    {
        fwrite((base_c + i), 1, 1, fp);
    }
}
static int decode_simple(MpiDecLoopData *data)
{
    RK_U32 pkt_done = 0;
    RK_U32 pkt_eos  = 0;
    RK_U32 err_info = 0;
    MPP_RET ret = MPP_OK;
    MppCtx ctx  = data->ctx;
    MppApi *mpi = data->mpi;
```

```
    char   *buf = data->buf;

    MppPacket packet = data->packet;

    MppFrame  frame  = NULL;

    size_t read_size = fread(buf, 1, data->packet_size, data->fp_i
nput);

    if (read_size != data->packet_size || feof(data->fp_input))

    {

        printf("found last packet\n");

        data->eos = pkt_eos = 1;

    }

    mpp_packet_write(packet, 0, buf, read_size);

    mpp_packet_set_pos(packet, buf);

    mpp_packet_set_length(packet, read_size);

    if (pkt_eos)

    {

        mpp_packet_set_eos(packet);

    }

    do {

        if (!pkt_done)

        {

            ret = mpi->decode_put_packet(ctx, packet);

            if (MPP_OK == ret)
```

```
            {
                pkt_done = 1;
            }
        }
        do {
            RK_S32 get_frm = 0;
            RK_U32 frm_eos = 0;
            ret = mpi->decode_get_frame(ctx, &frame);
            if (frame)
            {
                if (mpp_frame_get_info_change(frame))
                {
                    RK_U32 width = mpp_frame_get_width(frame);
                    RK_U32 height = mpp_frame_get_height(frame);
                    RK_U32 hor_stride = mpp_frame_get_hor_stride(fr
ame);
                    RK_U32 ver_stride = mpp_frame_get_ver_stride(fr
ame);
                    RK_U32 buf_size = mpp_frame_get_buf_size(fram
e);
                    printf("decode_get_frame get info changed found
\n");
                    printf("decoder require buffer w:h [%d:%d] stri
de [%d:%d] buf_size %d", width, height, hor_stride, ver_stride, bu
f_size);
                    if (NULL == data->frm_grp)
```

```
                {
                    ret = mpp_buffer_group_get_internal(&data->f
rm_grp, MPP_BUFFER_TYPE_ION);
                    if (ret)
                    {
                        printf("get mpp buffer group failed ret
 %d\n", ret);
                        break;
                    }
                    ret = mpi->control(ctx, MPP_DEC_SET_EXT_BUF_
GROUP, data->frm_grp);
                    if (ret)
                    {
                        printf("set buffer group failed ret %d\n
", ret);
                        break;
                    }
                }
                else
                {
                    ret = mpp_buffer_group_clear(data->frm_grp);
                    if (ret)
                    {
                        printf("clear buffer group failed ret %d
```

```
\n", ret);
```



```
                            break;

                        }

                    }

                    ret = mpp_buffer_group_limit_config(data->frm_g
rp, buf_size, 24);

                    if (ret)

                    {

                        printf("limit buffer group failed ret %d\n",
 ret);

                        break;

                    }

                    ret = mpi->control(ctx, MPP_DEC_SET_INFO_CHANGE
_READY, NULL);

                    if (ret)

                    {

                        printf("info change ready failed ret %d\n",
ret);

                        break;

                    }

                }

                else

                {

                    err_info = mpp_frame_get_errinfo(frame) | mpp_f
rame_get_discard(frame);

                    if (err_info)
```

```
                if (err_info)
                {
                    printf("decoder_get_frame get err info:%d di
scard:%d.\n", mpp_frame_get_errinfo(frame), mpp_frame_get_discard
(frame));
                }
                data->frame_count++;
                printf("decode_get_frame get frame %d\n", data
->frame_count);
                if (data->fp_output && !err_info)
                {
                    dump_mpp_frame_to_file(frame, data->fp_outp
ut);
                }
            }
            frm_eos = mpp_frame_get_eos(frame);
            mpp_frame_deinit(&frame);
            frame = NULL;
            get_frm = 1;
        }
        if (data->frm_grp)
        {
            size_t usage = mpp_buffer_group_usage(data->frm_gr
p);
            if (usage > data->max_usage)
            {
```

```
            {
```



```
                data->max_usage = usage;
            }
        }
        if (pkt_eos && pkt_done && !frm_eos)
        {
            msleep(10);
            continue;
        }
        if (frm_eos)
        {
            printf("found last frame\n");
            break;
        }
        if (data->frame_num && data->frame_count >= data->frame
_num)
        {
            data->eos = 1;
            break;
        }
        if (get_frm)
        {
            continue;
```

```
            }
```



```
            break;
        } while (1);
        if (data->frame_num && data->frame_count >= data->frame_nu
m)
        {
            data->eos = 1;
            printf("reach max frame number %d\n", data->frame_coun
t);
            break;
        }
        if (pkt_done)
        {
            break;
        }
    } while (1);
    return ret;
}
int mpi_dec_test_decode(MpiDecTestCmd *cmd)
{
    MPP_RET ret         = MPP_OK;
    size_t file_size    = 0;
    MppCtx ctx          = NULL;
    MppApi *mpi         = NULL;
```

```
MppPacket packet    = NULL;
MppFrame  frame     = NULL;
MpiCmd mpi_cmd      = MPP_CMD_BASE;
MppParam param      = NULL;
RK_U32 need_split   = 1;
RK_U32 width        = cmd->width;
RK_U32 height       = cmd->height;
MppCodingType type  = cmd->type;
char *buf           = NULL;
size_t packet_size  = MPI_DEC_STREAM_SIZE;
MppBuffer pkt_buf   = NULL;
MppBuffer frm_buf   = NULL;


MpiDecLoopData data;
printf("mpi_dec_test start\n");
memset(&data, 0, sizeof(data));
data.fp_input = fopen("test.h264", "rb");
if (NULL == data.fp_input)
{
    printf("failed to open input file %s\n", cmd->file_input);
    goto MPP_TEST_OUT;
}
```

```
    fseek(data.fp_input, 0L, SEEK_END);

    file_size = ftell(data.fp_input);

    rewind(data.fp_input);

    printf("input file size %ld\n", file_size);

    data.fp_output = fopen("output.yuv", "w+b");

    if (cmd->simple)

    {

        buf = mpp_malloc(char, packet_size);

        ret = mpp_packet_init(&packet, buf, packet_size);

    }

    printf("mpi_dec_test decoder test start w %d h %d type %d\n",
width, height, type);

    ret = mpp_create(&ctx, &mpi);

    mpi_cmd = MPP_DEC_SET_PARSER_SPLIT_MODE;

    param = &need_split;

    ret = mpi->control(ctx, mpi_cmd, param);

    if (MPP_OK != ret)

    {

        printf("mpi->control failed\n");

        goto MPP_TEST_OUT;

    }

    ret = mpp_init(ctx, MPP_CTX_DEC, type);
```

```
if (MPP_OK != ret)
```



```
{
    printf("mpp_init failed\n");
    goto MPP_TEST_OUT;
}
printf("packet_size = %d\n", packet_size);

data.ctx           = ctx;
data.mpi           = mpi;
data.eos           = 0;
data.buf           = buf;
data.packet        = packet;
data.packet_size    = packet_size;
data.frame         = frame;
data.frame_count    = 0;
data.frame_num      = cmd->frame_num;
printf("data.packet_size = %d\n", data.packet_size);
if (cmd->simple)
{
    while (!data.eos)
    {
        decode_simple(&data);
        printf("data.eos = %d\n", data.eos);
```

```
        }
    }
    cmd->max_usage = data.max_usage;
    ret = mpi->reset(ctx);
    if (MPP_OK != ret)
    {
        printf("mpi->reset failed\n");
        goto MPP_TEST_OUT;
    }
MPP_TEST_OUT:
    if (packet)
    {
        mpp_packet_deinit(&packet);
        packet = NULL;
    }
    if (frame)
    {
        mpp_frame_deinit(&frame);
        frame = NULL;
    }
    if (ctx)
    {
```

```
    mpp_destroy(ctx);
    ctx = NULL;
}
if (cmd->simple)
{
    if (buf)
    {
        mpp_free(buf);
        buf = NULL;
    }
}
if (data.pkt_grp)
{
    mpp_buffer_group_put(data.pkt_grp);
    data.pkt_grp = NULL;
}
if (data.frm_grp)
{
    mpp_buffer_group_put(data.frm_grp);
    data.frm_grp = NULL;
}
if (data.fp_output)
```

```
    {
        fclose(data.fp_output);
        data.fp_output = NULL;
    }
    if (data.fp_input)
    {
        fclose(data.fp_input);
        data.fp_input = NULL;
    }
    return ret;
}
int main(int argc, char **argv)
{
    RK_S32 ret = 0;
    MpiDecTestCmd  cmd_ctx;
    MpiDecTestCmd* cmd = &cmd_ctx;
    cmd->simple = 1;
    cmd->type = 7;
    cmd->width = 640;
    cmd->height = 480;
    ret = mpi_dec_test_decode(cmd);
    return ret;
```

```
}
```

### 6.1.4 compile demo:

```
$ gcc dec_mpp_test.c -o mpp_dec_test -lrockchip_mpp
```

### 6.1.4 run demo and see output:

```
$ ./mpp_dec_test

mpi_dec_test start

input file size 174123

mpi_dec_test decoder test start w 640 h 480 type 7

mpp[3589]: mpp_info: mpp version: 879c161 author: Xinhuang Li    2
020-09-15 [jpege]: change the upper limit of bps setting to 500Mbp
s

mpp[3589]: mpp_rt: NOT found ion allocator

mpp[3589]: mpp_rt: found drm allocator

mpp[3589]: mpp_log: got the /dev/mpp_service

packet_size = 4096

data.packet_size = 4096

data.eos = 0

data.eos = 0

data.eos = 0

data.eos = 0

decode_get_frame get info changed found
```

```
decoder require buffer w:h [176:144] stride [176:144] buf_size 506
88data.eos = 0

data.eos = 0

decode_get_frame get frame 1

base_length = 50688

decode_get_frame get frame 2

base_length = 50688

decode_get_frame get frame 3

base_length = 50688

decode_get_frame get frame 4

base_length = 50688

decode_get_frame get frame 5

base_length = 50688

decode_get_frame get frame 6

......

decode_get_frame get frame 820

base_length = 50688

decode_get_frame get frame 821

base_length = 50688

decode_get_frame get frame 822

base_length = 50688

decode_get_frame get frame 823
```

```
base_length = 50688
```



```
decode_get_frame get frame 824

base_length = 50688

decode_get_frame get frame 825

base_length = 50688

decode_get_frame get frame 826

base_length = 50688

decode_get_frame get frame 827

base_length = 50688

decoder_get_frame get err info:1 discard:0.

decode_get_frame get frame 828

found last frame

data.eos = 1

$ ls

mpp_dec_test  output.yuv  test.h264
```

**ps :**

Put a video of test.h264 in the current test directory.It will generate output.yuv.

## 6.2 Simple npu c++ demo

### 6.1.4 npu c++ demo source code and analysis:

```
//main.cc

#include <stdio.h>

#include <stdint.h>
```

```
#include <stdlib.h>

#include <fstream>

#include <iostream>

#include <sys/time.h>



#include "opencv2/core/core.hpp"

#include "opencv2/imgproc.hpp"

#include "opencv2/imgcodecs.hpp"



#include "rknn_runtime.h"



#include "ssd.h"



using namespace std;

using namespace cv;



/*-------------------------------------------

                Functions

-------------------------------------------*/



static void printRKNNTensor(rknn_tensor_attr *attr) {

    printf("index=%d name=%s n_dims=%d dims=[%d %d %d %d] n_elems
```

```
=%d size=%d fmt=%d type=%d qnt_type=%d fl=%d zp=%d scale=%f\n",
```



```
           attr->index, attr->name, attr->n_dims, attr->dims[3], a
ttr->dims[2], attr->dims[1], attr->dims[0],

           attr->n_elems, attr->size, 0, attr->type, attr->qnt_typ
e, attr->fl, attr->zp, attr->scale);

}



static unsigned char *load_model(const char *filename, int *model_
size)

{

    FILE *fp = fopen(filename, "rb");

    if(fp == nullptr) {

        printf("fopen %s fail!\n", filename);

        return NULL;

    }

    fseek(fp, 0, SEEK_END);

    int model_len = ftell(fp);

    unsigned char *model = (unsigned char*)malloc(model_len);

    fseek(fp, 0, SEEK_SET);

    if(model_len != fread(model, 1, model_len, fp)) {

        printf("fread %s fail!\n", filename);

        free(model);

        return NULL;

    }
```

```
    *model_size = model_len;
```



```
    if(fp) {
        fclose(fp);
    }
    return model;
}

/*-------------------------------------------
                Main Function
-------------------------------------------*/
int main(int argc, char** argv)
{
    const int img_width = 300;
    const int img_height = 300;
    const int img_channels = 3;

    rknn_context ctx;
    int ret;
    int model_len = 0;
    unsigned char *model;

    const char *model_path = argv[1];
    const char *img_path = argv[2];
```

```
    if (argc != 3) {
        printf("Usage:%s model image\n", argv[0]);
        return -1;
    }

    // Load image
    cv::Mat orig_img = cv::imread(img_path, 1);
    cv::Mat img = orig_img.clone();
    if(!orig_img.data) {
        printf("cv::imread %s fail!\n", img_path);
        return -1;
    }
    if(orig_img.cols != img_width || orig_img.rows != img_height)
{
        printf("resize %d %d to %d %d\n", orig_img.cols, orig_img.r
ows, img_width, img_height);
        cv::resize(orig_img, img, cv::Size(img_width, img_height),
 (0, 0), (0, 0), cv::INTER_LINEAR);
    }

    // Load RKNN Model
    printf("Loading model ...\n");
```

```
    model = load_model(model_path, &model_len);
```



```
    ret = rknn_init(&ctx, model, model_len, 0);
    if(ret < 0) {
        printf("rknn_init fail! ret=%d\n", ret);
        return -1;
    }

    // Get Model Input Output Info
    rknn_input_output_num io_num;
    ret = rknn_query(ctx, RKNN_QUERY_IN_OUT_NUM, &io_num, sizeof(io_num));
    if (ret != RKNN_SUCC) {
        printf("rknn_query fail! ret=%d\n", ret);
        return -1;
    }
    printf("model input num: %d, output num: %d\n", io_num.n_input, io_num.n_output);

    printf("input tensors:\n");
    rknn_tensor_attr input_attrs[io_num.n_input];
    memset(input_attrs, 0, sizeof(input_attrs));
    for (int i = 0; i < io_num.n_input; i++) {
        input_attrs[i].index = i;
        ret = rknn_query(ctx, RKNN_QUERY_INPUT_ATTR, &(input_attrs
```

```
[1]), sizeof(rknn_tensor_attr));
```



```
        if (ret != RKNN_SUCC) {
            printf("rknn_query fail! ret=%d\n", ret);
            return -1;
        }
        printRKNNTensor(&(input_attrs[i]));
    }

    printf("output tensors:\n");
    rknn_tensor_attr output_attrs[io_num.n_output];
    memset(output_attrs, 0, sizeof(output_attrs));
    for (int i = 0; i < io_num.n_output; i++) {
        output_attrs[i].index = i;
        ret = rknn_query(ctx, RKNN_QUERY_OUTPUT_ATTR, &(output_att
rs[i]), sizeof(rknn_tensor_attr));
        if (ret != RKNN_SUCC) {
            printf("rknn_query fail! ret=%d\n", ret);
            return -1;
        }
        printRKNNTensor(&(output_attrs[i]));
    }

    // Set Input Data
```

```
rknn_input inputs[1];
```



```
memset(inputs, 0, sizeof(inputs));
inputs[0].index = 0;
inputs[0].type = RKNN_TENSOR_UINT8;
inputs[0].size = img.cols*img.rows*img.channels();
inputs[0].fmt = RKNN_TENSOR_NHWC;
inputs[0].buf = img.data;


ret = rknn_inputs_set(ctx, io_num.n_input, inputs);
if(ret < 0) {
    printf("rknn_input_set fail! ret=%d\n", ret);
    return -1;
}


// Run
printf("rknn_run\n");
ret = rknn_run(ctx, nullptr);
if(ret < 0) {
    printf("rknn_run fail! ret=%d\n", ret);
    return -1;
}


// Get Output
```

```
    rknn_output outputs[2];

    memset(outputs, 0, sizeof(outputs));

    outputs[0].want_float = 1;

    outputs[1].want_float = 1;

    ret = rknn_outputs_get(ctx, io_num.n_output, outputs, NULL);

    if(ret < 0) {

        printf("rknn_outputs_get fail! ret=%d\n", ret);

        return -1;

    }



    // Post Process

    detect_result_group_t detect_result_group;

    postProcessSSD((float *)(outputs[0].buf), (float *)(outputs
[1].buf), orig_img.cols, orig_img.rows, &detect_result_group);

    // Release rknn_outputs

    rknn_outputs_release(ctx, 2, outputs);



    // Draw Objects

    for (int i = 0; i < detect_result_group.count; i++) {

        detect_result_t *det_result = &(detect_result_group.result
s[i]);

        printf("%s @ (%d %d %d %d) %f\n",

                det_result->name,
```

```
                det_result->box.left, det_result->box.top, det_resu
lt->box.right, det_result->box.bottom,

                det_result->prop);

        int x1 = det_result->box.left;

        int y1 = det_result->box.top;

        int x2 = det_result->box.right;

        int y2 = det_result->box.bottom;

        rectangle(orig_img, Point(x1, y1), Point(x2, y2), Scalar(2
55, 0, 0, 255), 3);

        putText(orig_img, det_result->name, Point(x1, y1 - 12), 1,
 2, Scalar(0, 255, 0, 255));

    }



    imwrite("./out.jpg", orig_img);



    // Release

    if(ctx >= 0) {

        rknn_destroy(ctx);

    }

    if(model) {

        free(model);

    }

    return 0;
```

```
}
```

### 6.1.4 compile demo:

```
//CMakeLists.txt

cmake_minimum_required(VERSION 3.4.1)



project(rknn_ssd_demo_linux)



set(CMAKE_SYSTEM_NAME Linux)



set(CMAKE_C_FLAGS "${CMAKE_C_FLAGS}")

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")



if (CMAKE_C_COMPILER MATCHES "aarch64")

  set(LIB_ARCH lib64)

else()

  set(LIB_ARCH lib)

endif()



# rknn api

#set(RKNN_API_PATH ${CMAKE_SOURCE_DIR}/../../librknn_api)

#include_directories(${RKNN_API_PATH}/include)
```

```
#set(RKNN_API_LIB ${RKNN_API_PATH}/${LIB_ARCH}/librknn_api.so)

set(RKNN_API_LIB /usr/lib/librknn_runtime.so)



# opencv

set(OpenCV_DIR /usr/share/OpenCV)

find_package(OpenCV REQUIRED)



set(CMAKE_INSTALL_RPATH "lib")



add_executable(rknn_ssd_demo

        src/main.cc

        src/ssd.cc

        )



target_link_libraries(rknn_ssd_demo

        ${RKNN_API_LIB}

        ${OpenCV_LIBS}

)



# install target and libraries

set(CMAKE_INSTALL_PREFIX ${CMAKE_SOURCE_DIR}/install/rknn_ssd_dem
o)
```

```
install(TARGETS rknn_ssd_demo DESTINATION ./)
```



```
install(DIRECTORY model DESTINATION ./)

install(PROGRAMS ${RKNN_API_LIB} DESTINATION lib)

$ make build

$ cd build

$ cmake ..

$ make

$ cp rknn_ssd_demo ../model/npu_test

$ cd ../model

$ ls

box_priors.txt  coco_labels_list.txt  npu_test  road.bmp  ssd_ince
ption_v2_rv1109_rv1126.rknn

$ mkdir model/

$ mv box_priors.txt  coco_labels_list.txt model/
```

### 6.1.4 run demo and see output:

```
$  ./npu_test ssd_inception_v2_rv1109_rv1126.rknn road.bmp

Loading model ...

model input num: 1, output num: 2

input tensors:

index=0 name= n_dims=4 dims=[1 300 300 3] n_elems=270000 size=2700
00 fmt=0 type=3 qnt_type=2 fl=127 zp=127 scale=0.007843

output tensors:

index=0 name= n_dims=4 dims=[1 1917 1 4] n_elems=7668 size=7668 fm
t=0 type=3 qnt_type=2 fl=-76 zp=180 scale=0.089482
```

```
index=1 name= n_dims=3 dims=[0 1 1917 91] n_elems=174447 size=1744
47 fmt=0 type=3 qnt_type=2 fl=-66 zp=190 scale=0.137463

rknn_run

loadLabelName

ssd - loadLabelName ./model/coco_labels_list.txt

loadBoxPriors

person @ (13 125 58 212) 0.984076

bicycle @ (171 165 278 234) 0.972723

person @ (110 119 152 197) 0.968828

person @ (206 113 256 216) 0.964399

car @ (146 133 217 170) 0.959365

person @ (83 134 92 158) 0.634101

person @ (49 133 58 156) 0.601661

person @ (96 134 105 162) 0.465688

$ ls

model  npu_test  out.jpg  rknn_ssd_demo  road.bmp  ssd_inception_v
2_rk180x.rknn  ssd_inception_v2_rv1109_rv1126.rknn
```

**ps :**

It will generate out.jpg.This image marks the object identified.