# Rockchip RK1808 Linux Docker Develop Guide

文件标识：RK-KF-YF-317

发布版本：V1.1.0

日期：2020-03-20

文件密级：公开资料



福州瑞芯微电子股份有限公司

Fuzhou 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

本文主要描述如何在RK1808 SDK上搭建Docker运行环境。

## 产品版本

| 芯片名称 | 内核版本 |
| --- | --- |
| RK1808 | Linux 4.4.185 |

## 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

| 日期 | 版本 | 作者 | 修改说明 |
| --- | --- | --- | --- |
| 2019-12-10 | V1.0.0 | Lin Jianhua | 初始版本 |
| 2020-03-20 | V1.1.0 | Lin Jianhua | 修改npu库路径 |

# 目录

# 1 kernel配置

按如下增减修改RK1808内核的默认配置。

```
arch/arm64/configs/rk1808_linux_defconfig
-# CONFIG_SWAP is not set
+CONFIG_POSIX_MQUEUE=y
+CONFIG_IKCONFIG=y
+CONFIG_IKCONFIG_PROC=y
+CONFIG_CGROUPS=y
+CONFIG_CGROUP_FREEZER=y
+CONFIG_CGROUP_PIDS=y
+CONFIG_CGROUP_DEVICE=y
+CONFIG_CPUSETS=y
+CONFIG_CGROUP_CPUACCT=y
+CONFIG_MEMCG=y
+CONFIG_MEMCG_SWAP=y
+CONFIG_MEMCG_KMEM=y
+CONFIG_CGROUP_HUGETLB=y
+CONFIG_CGROUP_PERF=y
+CONFIG_CGROUP_SCHED=y
+CONFIG_CFS_BANDWIDTH=y
+CONFIG_RT_GROUP_SCHED=y
+CONFIG_BLK_CGROUP=y
+CONFIG_NAMESPACES=y
+CONFIG_USER_NS=y
+CONFIG_BLK_DEV_THROTTLING=y
-# CONFIG_IOSCHED_CFQ is not set
+CONFIG_CFQ_GROUP_IOSCHED=y
+CONFIG_DEFAULT_NOOP=y
+CONFIG_SECCOMP=y
+CONFIG_XFRM_USER=y
-# CONFIG_INET_XFRM_MODE_TRANSPORT is not set
+CONFIG_INET_ESP=y
+CONFIG_NETFILTER=y
+CONFIG_BRIDGE_NETFILTER=y
+CONFIG_NF_CONNTRACK=y
+CONFIG_NF_CONNTRACK_FTP=y
+CONFIG_NF_CONNTRACK_TFTP=y
+CONFIG_NETFILTER_XT_MATCH_ADDRTYPE=y
+CONFIG_NETFILTER_XT_MATCH_CONNTRACK=y
+CONFIG_NETFILTER_XT_MATCH_IPVS=y
+CONFIG_IP_VS=y
+CONFIG_IP_VS_PROTO_TCP=y
+CONFIG_IP_VS_PROTO_UDP=y
+CONFIG_IP_VS_RR=y
+CONFIG_IP_VS_NFCT=y
+CONFIG_NF_CONNTRACK_IPV4=y
+CONFIG_IP_NF_IPTABLES=y
+CONFIG_IP_NF_FILTER=y
+CONFIG_IP_NF_NAT=y
+CONFIG_IP_NF_TARGET_MASQUERADE=y
+CONFIG_IP_NF_TARGET_REDIRECT=y
+CONFIG_BRIDGE=y
+CONFIG_BRIDGE_VLAN_FILTERING=y
+CONFIG_VLAN_8021Q=y
```

```
+CONFIG_NET_SCHED=y
+CONFIG_NET_CLS_CGROUP=y
+CONFIG_CGROUP_NET_PRIO=y
+CONFIG_CFG80211=y
+CONFIG_MAC80211=y
+CONFIG_MD=y
+CONFIG_BLK_DEV_DM=y
+CONFIG_DM_THIN_PROVISIONING=y
-# CONFIG_NET_CORE is not set
+CONFIG_DUMMY=y
+CONFIG_MACVLAN=y
+CONFIG_IPVLAN=y
+CONFIG_VXLAN=y
+CONFIG_VETH=y
+CONFIG_USB_USBNET=y
+CONFIG_USB_NET_RNDIS_HOST=y
+CONFIG_DEVPTS_MULTIPLE_INSTANCES=y
+CONFIG_EXT4_FS_POSIX_ACL=y
+CONFIG_EXT4_FS_SECURITY=y
+CONFIG_BTRFS_FS=y
+CONFIG_BTRFS_FS_POSIX_ACL=y
+CONFIG_OVERLAY_FS=y
+CONFIG_HUGETLBFS=y
+CONFIG_KEYS=y
```

利用docker官网提供check-config.sh脚本，check内核是否打开所有需要的配置。

```
adb push check-config.sh /data/
adb shell chmod 777 /data/check-config.sh
```

```
[root@rk1808:/]# /data/check-config.sh
info: reading kernel config from /proc/config.gz ...

Generally Necessary:
- cgroup hierarchy: properly mounted [/sys/fs/cgroup]
- CONFIG_NAMESPACES: enabled
- CONFIG_NET_NS: enabled
- CONFIG_PID_NS: enabled
- CONFIG_IPC_NS: enabled
- CONFIG_UTS_NS: enabled
- CONFIG_CGROUPS: enabled
- CONFIG_CGROUP_CPUACCT: enabled
- CONFIG_CGROUP_DEVICE: enabled
- CONFIG_CGROUP_FREEZER: enabled
- CONFIG_CGROUP_SCHED: enabled
- CONFIG_CPUSETS: enabled
- CONFIG_MEMCG: enabled
- CONFIG_KEYS: enabled
- CONFIG_VETH: enabled
- CONFIG_BRIDGE: enabled
- CONFIG_BRIDGE_NETFILTER: enabled
- CONFIG_NF_NAT_IPV4: enabled
- CONFIG_IP_NF_FILTER: enabled
- CONFIG_IP_NF_TARGET_MASQUERADE: enabled
- CONFIG_NETFILTER_XT_MATCH_ADDRTYPE: enabled
- CONFIG_NETFILTER_XT_MATCH_CONNTRACK: enabled
```

```
- CONFIG_NETFILTER_XT_MATCH_IPVS: enabled
- CONFIG_IP_NF_NAT: enabled
- CONFIG_NF_NAT: enabled
- CONFIG_NF_NAT_NEEDED: enabled
- CONFIG_POSIX_MQUEUE: enabled
- CONFIG_DEVPTS_MULTIPLE_INSTANCES: enabled

Optional Features:
- CONFIG_USER_NS: enabled
- CONFIG_SECCOMP: enabled
- CONFIG_CGROUP_PIDS: enabled
- CONFIG_MEMCG_SWAP: enabled
- CONFIG_MEMCG_SWAP_ENABLED: enabled
    (cgroup swap accounting is currently enabled)
- CONFIG_MEMCG_KMEM: enabled
- CONFIG_BLK_CGROUP: enabled
- CONFIG_BLK_DEV_THROTTLING: enabled
- CONFIG_IOSCHED_CFQ: enabled
- CONFIG_CFQ_GROUP_IOSCHED: enabled
- CONFIG_CGROUP_PERF: enabled
- CONFIG_CGROUP_HUGETLB: enabled
- CONFIG_NET_CLS_CGROUP: enabled
- CONFIG_CGROUP_NET_PRIO: enabled
- CONFIG_CFS_BANDWIDTH: enabled
- CONFIG_FAIR_GROUP_SCHED: enabled
- CONFIG_RT_GROUP_SCHED: enabled
- CONFIG_IP_NF_TARGET_REDIRECT: enabled
- CONFIG_IP_VS: enabled
- CONFIG_IP_VS_NFCT: enabled
- CONFIG_IP_VS_PROTO_TCP: enabled
- CONFIG_IP_VS_PROTO_UDP: enabled
- CONFIG_IP_VS_RR: enabled
- CONFIG_EXT4_FS: enabled
- CONFIG_EXT4_FS_POSIX_ACL: enabled
- CONFIG_EXT4_FS_SECURITY: enabled
- Network Drivers:
  - "overlay":
    - CONFIG_VXLAN: enabled
    - CONFIG_BRIDGE_VLAN_FILTERING: enabled
      Optional (for encrypted networks):
      - CONFIG_CRYPTO: enabled
      - CONFIG_CRYPTO_AEAD: enabled
      - CONFIG_CRYPTO_GCM: enabled
      - CONFIG_CRYPTO_SEQIV: enabled
      - CONFIG_CRYPTO_GHASH: enabled
      - CONFIG_XFRM: enabled
      - CONFIG_XFRM_USER: enabled
      - CONFIG_XFRM_ALGO: enabled
      - CONFIG_INET_ESP: enabled
      - CONFIG_INET_XFRM_MODE_TRANSPORT: enabled
  - "ipvlan":
    - CONFIG_IPVLAN: enabled
  - "macvlan":
    - CONFIG_MACVLAN: enabled
    - CONFIG_DUMMY: enabled
  - "ftp,tftp client in container":
    - CONFIG_NF_NAT_FTP: enabled
    - CONFIG_NF_CONNTRACK_FTP: enabled
```

```
    - CONFIG_NF_NAT_TFTP: enabled
    - CONFIG_NF_CONNTRACK_TFTP: enabled
- Storage Drivers:
  - "aufs":
    - CONFIG_AUFS_FS: missing
  - "btrfs":
    - CONFIG_BTRFS_FS: enabled
    - CONFIG_BTRFS_FS_POSIX_ACL: enabled
  - "devicemapper":
    - CONFIG_BLK_DEV_DM: enabled
    - CONFIG_DM_THIN_PROVISIONING: enabled
  - "overlay":
    - CONFIG_OVERLAY_FS: enabled
  - "zfs":
    - /dev/zfs: missing
    - zfs command: missing
    - zpool command: missing

Limits:
- /proc/sys/kernel/keys/root_maxkeys: 100000
```

脚本下载：$ curl https://raw.githubusercontent.com/docker/docker/master/contrib/check-config.sh > check-config.sh

# 2 Buildroot配置

文件系统按如下增加第三方包：

```
configs/rockchip_rk1808_defconfig
+BR2_PACKAGE_GIT=y
+BR2_PACKAGE_XORG7=y
+BR2_PACKAGE_XLIB_LIBX11=y
+BR2_PACKAGE_XZ=y
+BR2_PACKAGE_IPTABLES=y
+BR2_PACKAGE_CA_CERTIFICATES=y
+BR2_PACKAGE_DOCKER_ENGINE=y
+BR2_PACKAGE_DOCKER_ENGINE_EXPERIMENTAL=y
+BR2_PACKAGE_DOCKER_ENGINE_STATIC_CLIENT=y
+BR2_PACKAGE_DOCKER_ENGINE_DRIVER_BTRFS=y
+BR2_PACKAGE_DOCKER_ENGINE_DRIVER_DEVICEMAPPER=y
+BR2_PACKAGE_DOCKER_ENGINE_DRIVER_VFS=y
```

Buildroot默认带的docker 版本是17.05.0-ce, 如果需要最新版本的docker，请从https://download.docker.com/linux/static/stable/aarch64/下载，把解压后的bin文件推到开发板的/usr/bin目录下并赋予可执行权限。

详情参考官网：https://docs.docker.com/install/linux/docker-ce/binaries/#prerequisites

注意：由于增加这些资源包，编译出来的rootfs可能超过设定的分区大小（目前parameter中默认设置rootfs分区大小为1.5G），那么需要修改rootfs分区的大小。

```
ljh@SYS3:~/1808/releasex/device/rockchip$ git diff
diff --git a/rk1808/parameter-buildroot.txt b/rk1808/parameter-buildroot.txt
index 037c9a8..37f4829 100644
--- a/rk1808/parameter-buildroot.txt
+++ b/rk1808/parameter-buildroot.txt
@@ -8,5 +8,5 @@ MACHINE: 1808
 CHECK_MASK: 0x80
 PWR_HLD: 0,0,A,0,1
 TYPE: GPT
-CMDLINE:
mtdparts=rk29xxnand:0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00002000@0x00008000(misc),0x00010000@0x0000a000(boot),0x00010000@0x0001a000(recovery),0x00010000@0x0002a000(backup),0x00020000@0x0003a000(oem),0x00300000@0x0005a000(rootfs),-@0x0035a000(userdata:grow)
+CMDLINE:
mtdparts=rk29xxnand:0x00002000@0x00004000(uboot),0x00002000@0x00006000(trust),0x00002000@0x00008000(misc),0x00010000@0x0000a000(boot),0x00010000@0x0001a000(recovery),0x00010000@0x0002a000(backup),0x00020000@0x0003a000(oem),0x00400000@0x0005a000(rootfs),-@0x0045a000(userdata:grow)
```

# 3 运行hello-world镜像

1、挂载cgroupfs

从https://github.com/tianon/cgroupfs-mount下载挂载脚本，然后把cgroupfs-mount重命名为S70cgroupfs-mount，推到开发板的/etc/init.d目录下并赋予可执行权限。

```
adb push S70cgroupfs-mount /etc/init.d/
adb shell chmod a+x /etc/init.d/S70cgroupfs-mount
adb shell reboot
```

```
[root@rk1808:/]# mount
/dev/mmcblk2p8 on / type ext4 (rw,relatime,data=ordered)
devtmpfs on /dev type devtmpfs
(rw,relatime,size=1022236k,nr_inodes=255559,mode=755)
proc on /proc type proc (rw,relatime)
devpts on /dev/pts type devpts (rw,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /dev/shm type tmpfs
(rw,relatime,size=1022428k,nr_inodes=255607,mode=777)
tmpfs on /tmp type tmpfs (rw,relatime,size=1022428k,nr_inodes=255607)
tmpfs on /run type tmpfs
(rw,nosuid,nodev,relatime,size=1022428k,nr_inodes=255607,mode=755)
sysfs on /sys type sysfs (rw,relatime)
debug on /sys/kernel/debug type debugfs (rw,relatime)
pstore on /sys/fs/pstore type pstore (rw,relatime)
/dev/mmcblk2p7 on /oem type ext2 (rw,relatime)
/dev/mmcblk2p9 on /userdata type ext2 (rw,relatime)
none on /sys/kernel/config type configfs (rw,relatime)
adb on /dev/usb-ffs/adb type functionfs (rw,relatime)
cgroup on /sys/fs/cgroup type tmpfs
(rw,relatime,size=1022428k,nr_inodes=255607,mode=755)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,relatime,cpuset)
cgroup on /sys/fs/cgroup/cpu type cgroup (rw,relatime,cpu)
cgroup on /sys/fs/cgroup/cpuacct type cgroup (rw,relatime,cpuacct)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,relatime,blkio)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,relatime,memory)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,relatime,devices)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,relatime,freezer)
cgroup on /sys/fs/cgroup/net_cls type cgroup (rw,relatime,net_cls)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,relatime,perf_event)
cgroup on /sys/fs/cgroup/net_prio type cgroup (rw,relatime,net_prio)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,relatime,hugetlb)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,relatime,pids)
```

2、配置系统日期，不然在运行 docker run时会报certificate has expired错误

```
#date -s xxxx-xx-xx
```

3、启动docker守护进程

```
[root@rk1808:/]# dockerd&
[root@rk1808:/]# INFO[0000] libcontainerd: new containerd process, pid: 705
WARN[0000] containerd: low RLIMIT_NOFILE changing to max  current=1024
max=4096
```

```
ERRO[0001] Failed to built-in GetDriver graph zfs /var/lib/docker
INFO[0001] Graph migration to content-addressability took 0.00 seconds
INFO[0001] Loading containers: start.
WARN[0001] Running modprobe nf_nat failed with message: `modprobe: can't change directory to '/lib/modules': No such file or directory`, error: exit status 1
WARN[0001] Running modprobe xt_conntrack failed with message: `modprobe: can't change directory to '/lib/modules': No such file or directory`, error: exit status 1
WARN[0001] Could not load necessary modules for Conntrack: Running modprobe nf_conntrack failed with message: `modprobe: can't change directory to '/lib/modules': No such file or directory`, error: exit status 1
INFO[0001] Default bridge (docker0) is assigned with an IP address 172.17.0.0/16. Daemon option --bip can be used to set a preferred IP address
[  241.882816] IPv6: ADDRCONF(NETDEV_UP): docker0: link is not ready
INFO[0001] Loading containers: done.
WARN[0001] failed to retrieve docker-runc version: unknown output format: runc version commit: 9c2d8d184e5da67c95d601382adf14862e4f2228
spec: 1.0.0-rc2-dev

WARN[0001] failed to retrieve docker-init version: exec: "docker-init": executable file not found in $PATH
INFO[0001] Daemon has completed initialization
INFO[0001] Docker daemon                                 commit=89658be graphdriver=overlay2 version=17.05.0-ce
INFO[0002] API listen on /var/run/docker.sock
```

4、查看docker基本信息

```
[root@rk1808:/]# docker info
WARN[0008] failed to retrieve docker-runc version: unknown output format: runc version commit: 9c2d8d184e5da67c95d601382adf14862e4f2228
spec: 1.0.0-rc2-dev

WARN[0008] failed to retrieve docker-init version: exec: "docker-init": executable file not found in $PATH
Containers: 0
 Running: 0
 Paused: 0
 Stopped: 0
Images: 0
Server Version: 17.05.0-ce
Storage Driver: overlay2
 Backing Filesystem: extfs
 Supports d_type: true
 Native Overlay Diff: true
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
 Volume: local
 Network: bridge host macvlan null overlay
Swarm: inactive
Runtimes: runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 9048e5e50717ea4497b757314bad98ea3763c145
runc version: N/A (expected: 9c2d8d184e5da67c95d601382adf14862e4f2228)
```

```
init version: N/A (expected: )
Kernel Version: 4.4.185
Operating System: Buildroot 2018.02-rc3
OSType: linux
Architecture: aarch64
CPUs: 2
Total Memory: 1.951GiB
Name: rk1808
ID: GCD2:TBDU:322U:4VKT:64DE:5G3N:5IWD:3GOW:INIA:U22E:2XWX:IIGF
Docker Root Dir: /var/lib/docker
Debug Mode (client): false
Debug Mode (server): false
Registry: https://index.docker.io/v1/
Experimental: false
Insecure Registries:
 127.0.0.0/8
Live Restore Enabled: false
```

5、运行hello-world镜像

```
#docker run hello-world
Unable to find image 'hello-world:latest' locally
WARN[0132] failed to retrieve docker-runc version: unknown output format: runc version commit: 9c2d8d184e5da67c95d601382adf14862e4f2228
spec: 1.0.0-rc2-dev

WARN[0132] failed to retrieve docker-init version: exec: "docker-init": executable file not found in $PATH
latest: Pulling from library/hello-world
be6e184261a6: Pull complete
Digest: sha256:4fe721ccc2e8dc7362278a29dc660d833570ec2682f4e4194f4ee23e415e1064
Status: Downloaded newer image for hello-world:latest
[ 2193.265651] device veth13a3975 entered promiscuous mode
[ 2193.266376] IPv6: ADDRCONF(NETDEV_UP): veth13a3975: link is not ready
[ 2193.404746] IPv6: ADDRCONF(NETDEV_CHANGE): veth13a3975: link becomes ready
[ 2193.404846] docker0: port 1(veth13a3975) entered forwarding state
[ 2193.404884] docker0: port 1(veth13a3975) entered forwarding state
[ 2193.405019] IPv6: ADDRCONF(NETDEV_CHANGE): docker0: link becomes ready
[ 2193.408274] IPVS: Creating netns size=1840 id=1
[ 2193.423401] cgroup: docker-runc (974) created nested cgroup for controller "memory" which has incomplete hierarchy support. Nested cgroups may change behavior in the future.
[ 2193.424275] cgroup: "memory" requires setting use_hierarchy to 1 on the root
[ 2193.618156] docker0: port 1(veth13a3975) entered disabled state
[ 2193.619383] eth0: renamed from veth1ab5109
[ 2193.626744] docker0: port 1(veth13a3975) entered forwarding state
[ 2193.626808] docker0: port 1(veth13a3975) entered forwarding state

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
```

```
    (arm64v8)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent
it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

# 4 docker运行npu demo示例

1、从网盘链接：https://eyun.baidu.com/s/3bqn3MSf下载rk1808_docker_v1.tar

2、adb push npu-demo/galcore.ko /lib/modules/ && adb shell chmod 777 /lib/modules/galcore.ko

3、adb push npu-demo/npu /data/npu

4、adb push rk1808_ubuntu1804_v1 /data/ && adb shell reboot

5、按上面第3节运行hello-world的步骤执行1~3。

6、cd /data/

7、docker load --input rk1808_ubuntu1804_v1.tar //导入本地镜像

```
[root@rk1808:/userdata]# docker load --input rk1808_ubuntu1804_v1.tar
41094c861223: Loading layer  58.92MB/58.92MB
41f1cb418e12: Loading layer  991.2kB/991.2kB
c49ce40427a4: Loading layer  15.87kB/15.87kB
255e91841dfc: Loading layer  3.584kB/3.584kB
40e15cc8d1d2: Loading layer  147.3MB/147.3MB
Loaded image: rk1808:v1
[root@rk1808:/userdata]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
    SIZE
rk1808              v1                  e248b570c815        23 minutes ago
   200MB
```

8、docker run -it --device /dev/galcore:/dev/galcore -v /userdata/npu:/npu rk1808:v1 bash

9、cd npu && chmod 777 rknn_inference run.sh && ./run.sh

说明：rk1808_ubuntu1804_v1基础镜像，该镜像部署了ubuntu18.04并安装了isc-dhcp-server、ssh和rsyslog。如何制作docker镜像请参考官网：https://docs.docker.com/get-started/part2/