OpenWRT LEDE SDK

RTD 1619

Driving IC Innovation

SDKRelease/Directories

- Host Environment
- Bootcode
- Image-Builder
- Kernel-Release
- OpenWRT-LEDE
- Packages
- Toolchain

Host Environment (Ubuntu 16.04)

- Ubuntu 16.04
  - dpkg --add-architecture i386
  - apt-get update
  - apt-get install git-core build-essential libssl-dev libncurses5-dev unzip gawk zlib1g-dev subversion curl wget file python gettext libxml-parser-perl zip kmod cpio automake libarchive-zip-perl libswitch-perl bsdmainutils gperf bc device-tree-compiler libc6:i386 libncurses5:i386 libstdc++6:i386 zlib1g:i386 libarchive-zip-perl

Host Environment (Docker File)

```
FROM ubuntu:16.04
RUN sed -i 's/archive.ubuntu/tw.archive.ubuntu/g' /etc/apt/sources.list
# Setup required packages for runner and OpenWRT compilation
RUN dpkg --add-architecture i386 \
&& apt-get update \
&& apt-get install -y locales \
git-core build-essential \
libssl-dev libncurses5-dev unzip gawk zlib1g-dev \
subversion curl wget file python gettext \
libxml-parser-perl zip kmod cpio \
libswitch-perl bsdmainutils time gperf \
libc6:i386 libncurses5:i386 libstdc++6:i386 zlib1g:i386 \
libarchive-zip-perl \
&& apt-get clean
RUN locale-gen en_US.UTF-8; mkdir -p /mnt/data
RUN git config --global user.name "$USER"
RUN git config --global user.email "$USER@realtek.com"
```

SDKRelease/Bootcode

- U-Boot64
- Support Storages, SPI
- Rescue System
  - Update Firmwares, {Kernel, DTBs, Rescue Root Filesystem, Bluecore.audio}.
  - Load Rescue System from {USB, tftp, Storage}
- Dual Firmware Boot
  - Two Set of Firmwares in Storage
    - Use sequence number to indicate latest version
  - Boot from Latest Version Firmware
- On-Chip Recovery Mode
- Read SDKRelease/Bootcode/Docs to get details

SDKRelease/ImageBuilder

- Build Realtek SoC Image file
  - install.img
- Customization
  - Specify Firmware Layout in Storage
    - By feeds.conf
  - Support customer's Root Filesystem
  - Easy to Set
    - Kernel Boot Argument
    - MAC Address
    - By Bootargs.conf
  - Read SDKRelease/Image-Builder/Docs to get details

SDKRelease/Kernel-Release

- Kernel-Release/linux-linaro-stable-4.4-162.tar.xz
- Kernel-Release/patches
  - RTD-1619 porting patches
- Kernel-Release/kernel
  - Apply patches to linux-linaro-stable-4.4-162.tar.xz
  - Use by OpenWRT-LEDE
  - defconfig
    - rtd16xx_lede_spi_defconfig
    - rtd16xx_lede_transconde_spi_defconfig

SDKRelease/Packages

- FFMpeg 3.2.4
- Paragon ufs 9.4.4
- RTD1619 Firmwares
  - Bluecore.audio

SDKRelease/OpenWRT-LEDE

- OpenWRT-LEDE rootfs
- OpenWRT Configs
  - 1619_kernel44_nas.config - for pure NAS application
  - 1619_kernel44_nas_transcode.config - for transcode NAS application
  - Target Profile
    - Mjolnir eMMC – install.img stored in eMMC flash for Mjolnir board
    - Mjolnir SPI – install.img stored in SPI NOR flash for Mjolnir board
    - Megingjord eMMC – install.img stored in eMMC flash for Megingjord board
    - Megingjord SPI – install.img stored in SPI NOR flash for Megingjord board
- How to build OpenWRT
  - Prepare kernel source
    - Unpack kernel 4.4.162 and merge all patches into Kernel-Release/kernel
  - Use Default Configuration(configs/1619_kernel44_nas.config)
    - Enter OpenWRT-LEDE/
    - cp configs/1619_kernel44_nas.config .config
    - make oldconfig; make
  - Install.img
    - bin/targets/realtek/rtd16xx-glibc/install.img

SDKRelease/Toolchain

- Use gcc-arm-8.2-2018.11 from https://developer.arm.com
- Run aarch64-linux-gnu.sh to get aarch64 toolchain for building Kernel stand long
- Run arm-linux-gnueabihf.sh to get 32bit toolchain for building utilities for rescue rootfs. (MUST)

Appendix – 1/2

- How to support booting initramfs(normal rootfs) ?

1. Reserve memory to place normal rootfs.

Add "/memreserve/ ROOTFS_NORMAL_START ROOTFS_NORMAL_SIZE;"

in Kernel-Release/kernel/arch/arm64/boot/dts/realtek/rtd16xx/rtd-1619-mmnas-megingjord-2GB.dts or rtd-1619-nas-megingjord-2GB.dts

2. Modify Kernel-Release/kernel/include/soc/realtek/memory.h "ROOTFS_NORMAL_SIZE" the same as booting rootfs size

```
/* 0x02200000 ~ 0x025fffff */
#ifdef CONFIG_RTK_VMX_ULTRA
#define ROOTFS_NORMAL_START (0x4BB00000)
#define ROOTFS_NORMAL_SIZE  (0x12C00000) //300MB
#else
#define ROOTFS_NORMAL_START (0x02200000)
#define ROOTFS_NORMAL_SIZE (0x00400000) //4MB
#endif
```

Appendix – 2/2

3. Refer to Image-Builder/rescue-rootfs/initramfs.sh to generate normal rootfs. Please take usage of lzma in path of Image-Builder/x86_bin/lzma to compress rootfs.

4. Load normal rootfs to 0x02200000 memory offset while boot-up.

- By USB

  fatload usb 0:1 normal.root.spi.cpio.gz_pad.img 0x02200000

- By tftp

  tftp 0x02200000 normal.root.spi.cpio.gz_pad.img