

**RK3399 Android 8.1**

Software User Guide

Rev.A

August 31, 2020

Revision history

|  |  |  |
| --- | --- | --- |
| Revision | Date | Description |
| A | 2020-08-31 | Initial version |
|  |  |  |

Contents

1. Rockchip RK3399 Platform

1.1 Key features

2. RK3399 board

2.1 System block diagram

2.2 Start the board

2.3 Required equipment

3. Work with Android 8.1

3.1 개발환경 셋업

3.2 크로스 컴파일러 설치

3.3 소스코드 저장소

3.4 Android8.1 소스코드 컴파일

3.5 시스템 업데이트

4. 커스터마이징

4.1 패치파일 적용

4.2 부팅로고 변경

4.3 디바이스 정보 변경

4.4 카메라 HAL 드라이버 수정

4.5 Pre-installed 앱 패키징

5. USB디바이스 데모 응용프로그램

5.1 usbCameraTest

5.2 usbDepthCameraTest

5.3 usbAHRSTest

5.4 AstraSample

6. 이슈

**1. Rockchip RK3399 Platform**

**1.1 Key features**

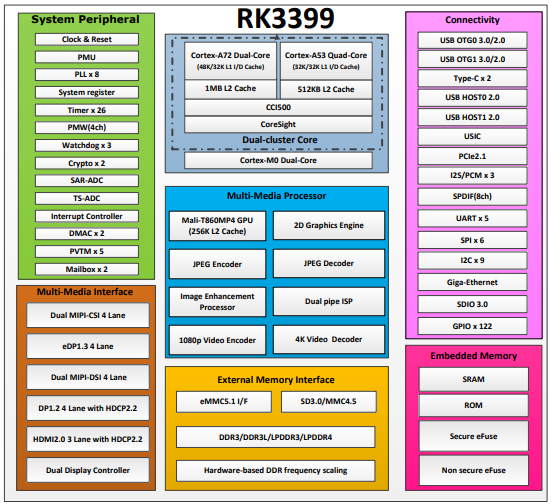


그림 1.1 RK3399 SoC Feature

RK3399는 Rockchip사의 SBC(Single Board Computer)용으로 출시된 Dual A72 + Quad A53 Cores와 Mali T860MP4 GPU를 탑재한 Application Processor입니다. 다음은 프로세서의 주요 특징을 정리한 것이다.

* SoC: Rockchip RK3399
  + CPU: big.LITTLE，Dual-Core Cortex-A72(up to 2.0GHz) + Quad-Core Cortex-A53(up to 1.5GHz)
  + GPU: Mali-T864 GPU，supports OpenGL ES1.1/2.0/3.0/3.1, OpenCL, DX11, and AFBC
  + VPU: 4K VP9 and 4K 10bits H265/H264 60fps decoding, Dual VOP, etc
* PMU: RK808-D PMIC, cooperated with independent DC/DC, enabling DVFS, sofeware power-down, RTC wake-up, system sleep mode
* RAM: Dual-Channel 4GB LPDDR3-1866
* Flash: 16GB~64GB eMMC 5.1 Flash
* Ethernet: Native Gigabit Ethernet
* Wi-Fi/BT: 802.11a/b/g/n/ac, Bluetooth 4.1, Wi-Fi and Bluetooth combo module, 2x2 MIMO, dual antenna interface
* Video Input: one or two 4-Lane MIPI-CSI, dual ISP, up to 13MPix/s，supports simultaneous input of dual camera data
* Video output
  + HDMI: HDMI 2.0a, supports 4K@60Hz，HDCP 1.4/2.2
  + DP on Type-C: DisplayPort 1.2 Alt Mode on USB Type-C
  + LCD Interface: one eDP 1.3（4-Lane，10.8Gbps）, one or two 4-Lane MIPI-DSI
* Audio Out: 3.5mm Dual channel headphone jack, or HDMI
* Audio In: 2-Pin 2.54mm microphone interface
* USB 2.0: 2 independent native USB 2.0 Host A interfaces
* USB 3.0: 1 native USB 3.0 Host A type interface
* USB Type-C: Supports USB3.0 Type-C and DisplayPort 1.2 Alt Mode on USB Type-C
* PCIe: One M.2 M-Key PCIe x4 socket, compatible with PCIe 2.1, Dual operation mode; Onboard M3 PCB nut for mounting M.2 2280 module
* microSD Slot x 1
* Keys: Power Key, Reset, MASKROM(BOOT), Recovery
* LED: 1 x power LED and 1 x GPIO Controlled LED
* IR receiver: Onboard IR receiver, Accepts 38KHz carrier frequency
* RTC Battery: 2 Pin 1.27/1.25mm RTC battery input connector
* Power supply: DC 12V/2A
* Ambient Operating Temperature: -20℃ to 70℃

**2. RK3399 board**

**2.1 System block diagram**

TBD

**2.2 Start the Board**

보드에 12V 전원 아답터를 연결하면 시스템이 자동 부팅이 된다. microSD에 부트로더 및 OS가 저장되어 있으면 SD부트모드로 진입한다. 일반적인 SD카드를 사용하면 내장 eMMC로 부팅이 되고 안드로이드가 시작한다.

HDMI 커넥터, USB포트 및 SD카드 슬롯에 해당 디바이스를 연결하면 해당 디바이스가 핫 플러그 방식으로 동작한다.

**2.3 Required equipment**

- LVRK3399 보드

- HDMI 커넥터 보드

- myAHRS+ 보드

- Astra Mini 센서 모듈

- UVC 웹카메라

- USB 2.0 & Type-C 케이블

- 12V DC 전원아답터

**3. Work with Android 8.1**

본 프로젝트에서는 안드로이드 8.1 운영체제를 사용하였다. 다음은 안드로이드 소스코드를 컴파일하고 새로운 이미지를 업데이트 하는 과정 및 시스템에 포함된 USB 디바이스 데모프로그램 등을 설명하였다.

**3.1 개발환경 셋업**

안드로이드 소스코드 컴파일을 위해서는 우분투 리눅스를 사용할 것을 권장하며, 하드디스크에 직접 설치, 가상머신 이미지를 통해 설치, Docker 및 Windows Subsystem for Linux(WSL)등의 환경에 설치가 가능하다. 하드디스크 용량은 최소 300GB 이상이 필요하고, 램은 최소 8GB, 가상머신의 경우 16GB 정도가 권장된다.

본 문서에서는 우분투 리눅스의 설치과정을 설명하지 않으며, 자세한 내용은 인터넷의 자료를 참고하도록 한다.

안드로이드 소스코드를 컴파일 하기위해 Ubuntu 16.04-64 bit 버전을 사용할 것을 권장하며, 다음의 패키지들을 설치한다.

|  |
| --- |
| sudo apt-get install bison g++-multilib git gperf libxml2-utils make python-networkx zip  sudo apt-get install flex curl libncurses5-dev libssl-dev zlib1g-dev gawk minicom  sudo apt-get install openjdk-8-jdk  sudo apt-get install exfat-fuse exfat-utils device-tree-compiler liblz4-tool |

**3.2 크로스 컴파일러 설치**

크로스 컴파일러는 리눅스 커널, u-boot 등을 컴파일하기위해 사용되며, 다음의 명령어를 통하여 설치가 가능하다.

|  |
| --- |
| git clone https://github.com/friendlyarm/prebuilts.git -b master --depth 1  cd prebuilts/gcc-x64  cat toolchain-6.4-aarch64.tar.gz\* | sudo tar xz -C / |

컴파일러가 설치된 디렉토리 경로를 ~/.bashrc 파일의 PATH변수에 추가한다.

|  |
| --- |
| **export** PATH=**/**opt**/**FriendlyARM**/**toolchain**/**6.4-aarch64**/**bin:$PATH  **export** GCC\_COLORS=auto |

추가한 사항을 유효하도록 ~/.bashrc 스크립트를 실행한다.

|  |
| --- |
| source ~/.bashrc |

컴파일러는 64비트용이고, 32비트 리눅스에서는 사용이 불가능하다. 다음 명령어를 사용하여 컴파일러가 설치가 잘 되었는지 확인한다.

|  |
| --- |
| aarch64-linux-gcc -v  Using built-in specs.  COLLECT\_GCC=aarch64-linux-gcc  COLLECT\_LTO\_WRAPPER=**/**opt**/**FriendlyARM**/**toolchain**/**6.4-aarch64**/**libexec**/**gcc**/**aarch64-cortexa53-linux-gnu**/**6.4.0**/**lto-wrapper  Target: aarch64-cortexa53-linux-gnu  Configured with: **/**work**/**toolchain**/**build**/**aarch64-cortexa53-linux-gnu**/**build**/**src**/**gcc**/**configure --build=x86\_64-build\_pc-linux-gnu  --host=x86\_64-build\_pc-linux-gnu --target=aarch64-cortexa53-linux-gnu --prefix=**/**opt**/**FriendlyARM**/**toolchain**/**6.4-aarch64  --with-sysroot=**/**opt**/**FriendlyARM**/**toolchain**/**6.4-aarch64**/**aarch64-cortexa53-linux-gnu**/**sysroot --enable-languages=c,**c++**  --enable-fix-cortex-a53-835769 --enable-fix-cortex-a53-843419 --with-cpu=cortex-a53  ...  Thread model: posix  **gcc** version 6.4.0 **(**ctng-1.23.0-150g-FA**)** |

**3.3 소스 저장소**

|  |  |
| --- | --- |
| **RK3399 Source Code Repositories** | |
| Android 8.1 Uboot | https://gitlab.com/friendlyelec/rk3399-android-8.1/tree/master/u-boot |
| Android 8.1 Linux-4.4.y Kernel | https://gitlab.com/friendlyelec/rk3399-android-8.1/tree/master/kernel |
| Android 8.1 BSP | https://gitlab.com/friendlyelec/rk3399-android-8.1 |

**3.4 Android 8.1 소스코드 컴파일**

안드로이드 소스코드는 다음의 두 가지 방법으로 설치가 가능하다.

* repo archive file on net-disk

|  |
| --- |
| <https://drive.google.com/file/d/1hMJgwDaXuyfTtbjiTFZpqEkX_pjIGH8Q/view?usp=sharing> |

위 경로에서 파일을 다운로드 받아 압축을 풀고, sync 스크립트를 실행하여 gitlab에서 최신 코드를 업데이트 한다.

|  |
| --- |
| tar zxvf rk3399-android-8.1.git-20181228.tgz  cd rk3399-android-8.1  ./sync.sh |

* git clone from gitlab

FriendlyElec 사의 RK3399 BSP 소스코드는 gitlab에서 유지 보수되고, 다음의 명령어를 통하여 다운로드가 가능하다.

|  |
| --- |
| **git clone** https:**//**gitlab.com**/**friendlyelec**/**rk3399-android-8.1 --depth 1 -b master |

다음 그림 3.1은 안드로이드 8.1 SDK의 디렉토리 구조이다.

|  |
| --- |
|  |

그림 3.1 안드로이드 8.1 SDK 디렉토리 구조

소스코드가 다운로드 되면, 다음과 같이 컴파일을 하고 이미지 파일을 생성한다.

|  |
| --- |
| cd rk3399-android-8.1  ./build-lvrk3399.sh -F -M |

-F 옵션은 build all (U-Boot, kernel, wifi, android)을 의미하고,

-M 옵션은 make rockdev image를 의미하여 각 이미지 파일들을 생성하고,

-a 옵션은 android, -B 옵션은 U-Boot, -K 옵션은 커널을 각각 빌드를 한다.

**3.5 시스템 업데이트**

컴파일이 완료되면 바이너리 이미지들을 다시 다운로드 해야 한다. 다음에 소개하는 EFlasher와 AndroidTool을 사용하는 방법 등이 있다.

릴리즈한 하드웨어에 업데이트할 바이너리 이미지 파일은 다음의 경로에서 다운로드 가능하다.

|  |
| --- |
| https://drive.google.com/file/d/1FMVh-t5OBZFF8xNdmzUbrALJn4lqz1uP/view?usp=sharing |

* EFlasher를 사용한 SD to eMMC 다운로드

|  |
| --- |
| <https://drive.google.com/file/d/1pJ9ZFOAHvwnkGVfujkiYMQ2dcFEXVJHe/view?usp=sharing> |

위 경로의 rk3399-eflasher-android8-YYYYMMDD.img.zip 파일을 다운로드 받아 압축을 풀고 img파일을 Win32 Disk Imager 유틸리티를 사용하여 EFlasher Tool을 SD카드(최소8GB)에 설치한다. 다음의 경로에서 해당 유틸리티를 다운로드 받는다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/tools/win32diskimager.rar |

그리고 3.4절에서 컴파일이 완료되면 “/rockdev/Image-\*\*\*/” 디렉토리에 생성된 파일들을 복사하여 그림 3.2과 같이 SD카드의 FriendlyARM 파티션의 android8 폴더안에 저장한다.

|  |
| --- |
|  |

그림 3.2 다운로드 이미지 파일

복사가 완료되면 SD카드를 RK3399 보드에 장착하고, 전원을 켜면 SD Boot모드로 진입하여 EFlasher가 실행되고, 그림 3.3와 같이 설치할 OS를 선택하여 eMMC에 설치한다.

|  |
| --- |
|  |

그림 3.3 EFlasher 실행 중, 설치 OS 선택 화면

* Android Tool을 사용하여 USB Type-C 통해 다운로드

다음의 경로를 통해 DriverAssistant\_v4.5.zip(RK3399’ USB Driver) 파일을 다운로드 받아 Windows PC에 설치한다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/tools/DriverAssitant\_v4.5.zip |

다음의 경로를 통해 AndroidTool\_2.54.zip 파일을 다운로드 받아 압축을 푼다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/tools/AndroidTool\_Release\_v2.54.zip |

3.4절에서 생성한 이미지 파일들을 AndroidTool\_Release\_v2.54 디렉토리 아래 그림과 같이 rockdev/image 디렉토리 위치에 저장한다.

아래 그림 3.4의 AndroidTool.exe 파일을 관리자 모드로 실행을 하고, 보드의 RECOVERY 버튼을 누르고 전원버튼이나 RST버튼을 누르고, 2초 후 전원버튼 혹은 RST버튼을 릴리즈 한다.

|  |  |
| --- | --- |
|  |  |

그림 3.4 AndroidTool 파일 디렉토리

그러면 그림 3.5의 좌측과 같이 앱의 하단부에 “Found One LOADER Device” 라고 인식이 되면 그림 3.4의 우측과 같이 “Run” 버튼을 눌러 다운로드를 한다. 다운로드가 성공적으로 완료되면 안드로이드가 자동으로 부팅이 된다.

|  |  |
| --- | --- |
|  |  |

그림 3.5 AndroidTool 이미지 다운로드

**4. 커스터마이징**

RK3399 레퍼런스 보드와 더불어 보드 제작회사에서 제공하는 BSP를 포함한 안드로이드 소스코드를 필요에 알맞도록 수정하여 사용할 수 있다.

2020-06-23에 릴리즈 된 안드로이드8.1 BSP를 Baseline으로 사용하였으며, Android8.1\_SDK를 Rockchip 공식버전 v8.32\_20200218에 업데이트 하였다. 다음은 최근에 업데이트된 사항이다.

|  |
| --- |
| **2020-06-23**  Provides Settings to automatically hide navigation bars  **2020-03-24**  1) Update Android8.1\_SDK to Rockchip official version v8.32\_20200218  2) Upgrade OpenCV to 4.2  3) Upgrade OpenWrt to 19.07.1  4) DDR loader updated to 1.24, support multiple frequency set points  5) Improved camera driver stability  6) Optimization of kernel stability  **2019-09-26 이전 사항은 생략** |

**4.1 패치파일 적용**

기본으로 제공되는 BSP에 LVRK3399\_Rev2 보드에 필요한 수정사항을 적용한 패치 파일을 사용하면 새로운 빌드 서버 셋업 시 편리하다.

패치파일 라인수가 8046줄이므로 지면관계상 일부만 아래에 참조하였다.

|  |
| --- |
| … 생략…  diff --git a/kernel/arch/arm64/boot/dts/rockchip/rk3399-nanopi4-common.dtsi b/kernel/arch/arm64/boot/dts/rockchip/rk3399-nanopi4-common.dtsi  index 30b8c95175..5878ec2a51 100644  --- a/kernel/arch/arm64/boot/dts/rockchip/rk3399-nanopi4-common.dtsi  +++ b/kernel/arch/arm64/boot/dts/rockchip/rk3399-nanopi4-common.dtsi  @@ -1002,8 +1002,8 @@  };    &sdio0 {  - clock-frequency = <150000000>;  - clock-freq-min-max = <100000 150000000>;  + clock-frequency = <50000000>;  + clock-freq-min-max = <100000 50000000>;  supports-sdio;  bus-width = <4>;  disable-wp;  diff --git a/kernel/logo.bmp b/kernel/logo.bmp  index 2a85f093d0..78390c0d01 100644  Binary files a/kernel/logo.bmp and b/kernel/logo.bmp differ  diff --git a/kernel/logo\_kernel.bmp b/kernel/logo\_kernel.bmp  index 2a85f093d0..78390c0d01 100644  Binary files a/kernel/logo\_kernel.bmp and b/kernel/logo\_kernel.bmp differ  … 이하 생략 … |

패치파일 적용방법은 다음과 같다.

다음의 경로에서 패치 파일을 다운로드 한다.

|  |
| --- |
| <https://github.com/keti-vrar/rk3399_android8.1//blob/master/lvrk3399_patch> |

패치는 lvrk3399\_patch.patch 파일을 프로젝트 폴더 바로 바깥에 저장해두고 실행해야 하며, 다음의 명령어를 커맨드라인에서 입력한다.

|  |
| --- |
| patch -p1 < ../lvrk3399\_patch.patch |

**4.2 부팅 로고 변경**

부팅 로고 이미지는 아래의 그림 4.1의 이미지를 800x480x24b bitmap 포맷으로 사용하였으며, 로고로 사용한 비트맵 이미지를 커널 디렉토리에 저장하고 다시 컴파일을 하여 커널 이미지를 만든다.

|  |
| --- |
|  |

그림 4.1 부팅 로고

**4.3 디바이스 정보 변경**

안드로이드 시스템 설정메뉴에서 필요에 따라 제품의 디바이스의 정보를 변경하였으며,

문서와 함께 제공하는 패치를 적용하여 디바이스 모델 정보를 그림 4.2와 같이 변경을 하였다.

|  |  |  |
| --- | --- | --- |
|  |  |  |

그림 4.2 디바이스 정보

디바이스의 정보 변경은 빌드 스크립트 및 코드레벨에서 수정을 하였으며, 자세한 사항은 아래의 패치 파일에 반영되어 있다.

|  |
| --- |
| Diff --git a/device/rockchip/rk3399/nanopc\_t4.mk b/device/rockchip/rk3399/nanopc\_t4.mk  index 374d0ad881..3961e42591 100644  --- a/device/rockchip/rk3399/nanopc\_t4.mk  +++ b/device/rockchip/rk3399/nanopc\_t4.mk  @@ -1,8 +1,4 @@  -#  -# Copyright 2014 The Android Open-Source Project  -#  -# Licensed under the Apache License, Version 2.0 (the "License");  -# you may not use this file except in compliance with the License.  +# # Copyright 2014 The Android Open-Source Project # # Licensed under the Apache License, Version 2.0 (the "License"); # you may not use this file except in compliance with the License.  # You may obtain a copy of the License at  #  # http://www.apache.org/licenses/LICENSE-2.0  @@ -32,8 +28,8 @@ PRODUCT\_CHARACTERISTICS := tablet  PRODUCT\_NAME := nanopc\_t4  PRODUCT\_DEVICE := nanopc-t4  PRODUCT\_BRAND := Android  -PRODUCT\_MODEL := NanoPC-T4 (RK3399)  -PRODUCT\_MANUFACTURER := FriendlyELEC (www.friendlyarm.com)  +PRODUCT\_MODEL := EA-MX50H  +PRODUCT\_MANUFACTURER := ElvisionTech (www.elvisiontech.com)    # Screen size is "normal", density is "hdpi"  PRODUCT\_AAPT\_CONFIG := normal large xlarge mdpi hdpi xhdpi  diff --git a/frameworks/base/core/java/android/os/Build.java b/frameworks/base/core/java/android/os/Build.java  index 062799891f..960be631f9 100644  --- a/frameworks/base/core/java/android/os/Build.java  +++ b/frameworks/base/core/java/android/os/Build.java  @@ -42,7 +42,8 @@ public class Build {  public static final String ID = getString("ro.build.id");    /\*\* A build ID string meant for displaying to the user \*/  - public static final String DISPLAY = getString("ro.build.display.id");  + //public static final String DISPLAY = getString("ro.build.display.id");  + public static final String DISPLAY = "EA-MX50H." + ID;    /\*\* The name of the overall product. \*/  public static final String PRODUCT = getString("ro.product.name"); |

**4.4 카메라 HAL 드라이버 수정**

Rockchip에서 릴리즈되는 BSP는 카메라 프리뷰가 저해상도 10fps대 이상을 지원하지 않는다. 따라서 카메라 HAL코드를 수정해줘야 한다. 수정사항은 패치파일에 포함되어 있으며 다음과 같다.

|  |
| --- |
| diff --git a/hardware/rockchip/camera/CameraHal/CameraIspAdapter.cpp b/hardware/rockchip/camera/CameraHal/CameraIspAdapter.cpp  old mode 100755  new mode 100644  index 0c9b207d45..7a1c330b5f  --- a/hardware/rockchip/camera/CameraHal/CameraIspAdapter.cpp  +++ b/hardware/rockchip/camera/CameraHal/CameraIspAdapter.cpp  @@ -394,7 +394,10 @@ status\_t CameraIspAdapter::startPreview(int preview\_w,int preview\_h,int w, int h  }    is\_video = (((preview\_w == 1920) && (preview\_h == 1080)) ||  - ((preview\_w == 1280) && (preview\_h == 720)));  + ((preview\_w == 1280) && (preview\_h == 720)) ||  + ((preview\_w == 800) && (preview\_h == 600)) ||  + ((preview\_w == 640) && (preview\_h == 480))  + );    //must to get illum befor resolution changed  if (is\_capture) {  @@ -2482,20 +2485,19 @@ void CameraIspAdapter::bufferCb( MediaBuffer\_t\* pMediaBuffer )  FramInfo\_s \*tmpFrame=(FramInfo\_s \*)malloc(sizeof(FramInfo\_s));  if(!tmpFrame){  MediaBufUnlockBuffer( pMediaBuffer );  - return;  - }  - //add to vector  - memset(tmpFrame, 0x0, sizeof(\*tmpFrame));  - tmpFrame->frame\_index = (ulong\_t)tmpFrame;  - tmpFrame->phy\_addr = (ulong\_t)phy\_addr;  - tmpFrame->frame\_width = width;  - tmpFrame->frame\_height= height;  - tmpFrame->vir\_addr = (ulong\_t)y\_addr\_vir;  - tmpFrame->frame\_fmt = fmt;  - tmpFrame->used\_flag = 1;  - tmpFrame->vir\_addr\_valid = true;  + return; }  + //add to vector  + memset(tmpFrame, 0x0, sizeof(\*tmpFrame));  + tmpFrame->frame\_index = (ulong\_t)tmpFrame;  + tmpFrame->phy\_addr = (ulong\_t)phy\_addr;  + tmpFrame->frame\_width = width;  + tmpFrame->frame\_height= height;  + tmpFrame->vir\_addr = (ulong\_t)y\_addr\_vir;  + tmpFrame->frame\_fmt = fmt;  + tmpFrame->used\_flag = 1;  + tmpFrame->vir\_addr\_valid = true;  #if (USE\_RGA\_TODO\_ZOOM == 1)  - tmpFrame->zoom\_value = mZoomVal;  + tmpFrame->zoom\_value = mZoomVal;  #else  if((tmpFrame->frame\_width > 2592) && (tmpFrame->frame\_height > 1944) && (mZoomVal != 100) ) {  tmpFrame->zoom\_value = mZoomVal;  diff --git a/hardware/rockchip/camera/CameraHal/CameraSocAdapter.cpp b/hardware/rockchip/camera/CameraHal/CameraSocAdapter.cpp  index fe7b1cc36e..0c55635251 100644  --- a/hardware/rockchip/camera/CameraHal/CameraSocAdapter.cpp  +++ b/hardware/rockchip/camera/CameraHal/CameraSocAdapter.cpp  @@ -197,12 +197,12 @@ int CameraSOCAdapter::cameraFpsInfoSet(CameraParameters &params)    parameterString.append("),(15000,15000),(24000,24000)");  #else  - params.set(CameraParameters::KEY\_PREVIEW\_FPS\_RANGE, "19000,19000");  - parameterString.append("(15000,15000),(19000,19000),(24000,24000)");  + params.set(CameraParameters::KEY\_PREVIEW\_FPS\_RANGE, "30000,30000");  + parameterString.append("(15000,15000),(19000,19000),(24000,24000),(30000,30000)");  #endif  memset(framerates,0x00,sizeof(framerates));  - strcpy(framerates,"15,19,24");  - params.setPreviewFrameRate(24);  + strcpy(framerates,"15,19,24,30");  + params.setPreviewFrameRate(30);  //parameterString.append(")");    params.set(CameraParameters::KEY\_SUPPORTED\_PREVIEW\_FPS\_RANGE, parameterString.string()); |

**4.5 Prebuilt-앱 패키징**

Pre-installed 앱은 공장초기화를 하여도 삭제되지 않고 기본으로 설치되어 있는 시스템 응용프로그램이다. 처음 디바이스를 시작하면 안드로이드 SDK8.1 기본 앱과 더불어 Google Play Store와 FriendlyElec사의 HW테스트 프로그램 몇 개가 기본으로 탑재되어 있는데, 불필요한 앱은 삭제하고 대신하여 다음의 5절에서 소개되는 USB 디바이스 테스트 프로그램 등을 시스템 앱으로 적용해 볼 수 있다.

이미지 패키지에 포함시킬 앱의 바이너리는 다음의 경로에 저장시킨다.

|  |
| --- |
| /$(ANDROID\_ROOT)/vendor/rockchip/common/apps/$(YOUR\_APPNAME)/($YOUR\_APPNAME).apk |

그리고 apps.mk파일을 다음과 같이 수정한다.

|  |
| --- |
| diff --git a/vendor/rockchip/common/apps/apps.mk b/vendor/rockchip/common/apps/apps.mk  index 621d331722..9ba5f24239 100755  --- a/vendor/rockchip/common/apps/apps.mk  +++ b/vendor/rockchip/common/apps/apps.mk  @@ -54,7 +54,9 @@ endif  ifeq ($(TARGET\_PRODUCT), nanopc\_t4)  PRODUCT\_PACKAGES += \  TfLiteCameraDemo \  - WifiDisplay  + WifiDisplay \  + USBCamera \  + USBImu  Endif |

**5. USB디바이스 데모 응용프로그램**

**5.1 usbCameraTest**

UVC 규격과 호환하는 카메라 데모 앱이다. 보드에 연결된 카메라가 다수인 경우 리스트에서 필요에 따라 선택하여 실행이 가능하다.

해상도는 1280 x 720 단일 해상도를 지원하며 USB2.0-HS 모드에서 30 Hz로 동작한다. 또한 스틸 이미지 캡쳐, 동영상 녹화 기능을 지원하고, 밝기, 대조 값이 실시간 조정 가능하다.

본 프로그램은 모든 안드로이드 장비와 호환되지 않으며, 프로젝트에 사용된 Rxsx Video UVC 장비와 RK3399 안드로이드 8.1 시스템 환경에서 구동이 된다.

APK 설치파일은 다음의 경로에서 다운로드 가능하다

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/apk/usbCameraTest.apk |

다음 그림 5.1은 프로그램 실행 화면이다.

|  |  |
| --- | --- |
|  |  |

그림 5.1 usbCameraTest 프로그램 실행 화면

**5.2 usbDepthCameraTest**

OpenNI(Natural Interaction) API와 호환되는 Depth 카메라를 구동하는 데모 앱이다. 프로젝트에서 사용하는 Orbbec사의 Astra Mini 시리즈의 디바이스가 구동이 가능하다.

APK 설치파일은 다음의 경로에서 다운로드 가능하다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/apk/usbDepthCameraTest.apk |

다음 그림 5.2는 프로그램 실행 화면이다.

|  |
| --- |
|  |

그림 5.2. usbDepthCameraTest 프로그램 실행 화면

**5.3 usbAHRSTest**

WITHROBOT사의 myAHRS+ 디바이스와 호환되는 IMU 모션 센서를 구동하는 데모 앱이다. 해당 제품에 사용된 칩은 TDK사의 9-axis Motion Tracking Device인 MPU-9250 이다.

APK 설치파일은 다음의 경로에서 다운로드 가능하다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/apk/usbAHRSTest.apk |

다음 그림 5.3은 프로그램 실행 화면이다.

|  |
| --- |
|  |

그림 5.3 usbAHRSTest 프로그램 실황 화면

**5.4 AstraSample**

Orbbec사에서 제공하는 UVC 및 OpenNI API와 호환되는 디바이스에서 실행하는 데모 앱이다.

ASTRA SDK2.1.1 호환이고 사용하는 USB 2.0 케이블 임피던스 상태에 민감하게 동작한다. 임피던스가 낮은 케이블을 사용하고, 2.4W 이하의 전력을 소비하므로 USB포트가 출력전류가 충분하지 못하면 동작이 원활하지 못하다.

APK 설치파일은 다음의 경로에서 다운로드 가능하다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/apk/astraMiniSampleTest.apk |

다음 그림 5.4는 프로그램 실행 화면이다.

|  |
| --- |
|  |

그림 5.4 Astra Sample 프로그램 실행 화면

**6. 이슈**

TBD

본 문서는 다음의 경로에서 최신 버전을 다운로드 할 수 있다.

|  |
| --- |
| https://github.com/keti-vrar/rk3399\_android8.1/blob/master/doc/RK3399\_Android\_8\_User%20Manual.docx |