

TCC803x Automotive Linux SDK

Quick Start Guide

**Rev. 2.10 [A]
2018-09-05**

※ The information in this document is subject to change without notice and should not be construed as a commitment by Telechips Inc.

Kindly visit <http://www.telechips.com> for more information.

© 2018 Telechips Inc. All rights reserved.

TABLE OF CONTENTS

Contents

1 Introduction.....	5
2 Board Description	6
2.1 TCC803x EVB	6
2.1.1 USB Mode Switch	7
2.1.2 SNOR Boot Mode Switch	7
2.1.3 Function Switch.....	8
3 Build Guide	9
3.1 SDK Build Preparation	9
3.2 Yocto Project	9
3.3 Task Process	9
3.4 Yocto Project Installation	10
3.4.1 Configuration for Yocto Project.....	10
3.4.2 Linux distribution versions supported by Yocto Project	10
3.4.3 List of package required for using Yocto Project	10
3.5 ALS Installation	11
3.5.1 repo Installation.....	11
3.5.2 Download ALS	11
3.5.3 Composition of ALS.....	11
3.5.4 Yocto Project buildtools Installation.....	12
3.6 Building ALS.....	13
3.6.1 Configuring the Yocto Project build environment.....	13
3.6.2 Bitbake Configuration.....	13
3.6.2.1 Configuration using Auto script.....	13
3.6.3 User definition Configuration.....	14
3.6.3.1 Setting Environmental Variable of Template	14
3.6.3.2 Bitbake Configuration	14
3.6.3.3 Setup Configuration.....	15
3.6.3.4 Setup Graphics System and Qt Platform Abstraction (QPA)	15
3.6.4 Build automotive-linux-platform-image	16
3.6.4.1 Build image type.....	16
3.6.4.1.1 telechips-ivi-image-minimal.....	16
3.6.4.1.2 telechips-ivi-image-gstreamer.....	16
3.6.4.1.3 telechips-ivi-image	16
3.6.4.1.4 automotive-linux-platform-image.....	16
3.6.4.2 Location of results	17
3.7 FAQs on Build	18
3.7.1 How to rebuild package.....	18
3.7.1.1 clean and build	18
3.7.1.2 cleanall and build.....	18
3.7.1.3 cleansstate and build.....	18
3.7.2 How to rebuild all packages	18
3.7.3 How to modify source code.....	18
3.7.4 How to add extra package to image.....	18
3.7.5 How to modify rootfs image without modify recipes	18
3.7.6 How to change from read-only to writable rootfs	19
3.7.7 How to enable network environment.....	19
3.7.8 How to enable ssh environment.....	19
3.7.9 How to install gdb environment	19
3.8 Building Application Development Toolkit (ADT).....	20
3.8.1 Bitbake Configuration.....	20
3.8.2 ADT type provided by ALS	20
3.8.2.1 meta-toolchain-telechips (Application Development Toolkit).....	20
3.8.2.2 meta-toolchain-telechips-ivi (Application Development Toolkit include GStreamer)	20
3.8.2.3 meta-toolchain-telechips-qt5 (Application Development Toolkit include GStreamer and Qt5).....	20
3.8.3 Build Application Development Toolkit (ADT)	21
3.8.4 Location of result.....	21
4 FWDN (Firmware DownloadER) Guide	22
4.1 System Requirement.....	22
4.1.1 Software Requirement	22
4.1.2 Hardware Requirement	22
4.2 ALS Firmware Construction	23
4.3 Install VTC Driver	24
4.4 ALS Firmware Download Sequence.....	24
4.4.1 Set Board to USB Mode.....	24
4.4.2 Set Board to SNOR Boot Mode	24
4.4.3 Add Bootloader (u-boot).....	24

4.4.4	Add Micom Image	25
4.4.5	Connect Board and USB cable	25
4.4.6	Configuring Partition.....	27
4.4.6.1	Configuring Partition of SD Data	27
4.4.6.1.1	SD Data Partition Information	30
4.4.6.2	Writing ALS Firmware.....	31
4.4.6.3	Firmware Write Option.....	32
4.4.6.3.1	Default Download	32
4.4.6.3.2	Low format whole memory before download	33
5	Partition Update.....	34
5.1	Fastboot Flash Memory update.....	34
5.1.1	What is Fastboot	34
5.1.2	Fastboot Mode	34
5.1.3	Installing a USB Driver	35
5.1.4	Fastboot flash memory update.....	35
6	Bootimg and Running Guide.....	36
6.1	Board Booting Guide	36
6.1.1	Bootimg Screen	36
6.1.2	Linux Console Login.....	36
6.2	System Architecture	37
6.2.1	Block Diagram.....	37
6.2.2	Structure	37
6.2.2.1	Linux Kernel Layer	37
6.2.2.2	Middleware	37
6.2.2.3	Daemon.....	38
6.2.2.4	Application.....	38
6.2.3	Motion Scenario	38
6.3	ALS Application Guide.....	39
6.3.1	Available Documentation	39
6.3.2	Launcher	40
6.3.2.1	Overview	40
6.3.2.2	Block Diagram	40
6.3.2.3	Feature	40
6.3.2.4	Required packages	40
6.3.2.5	UI Guide	40
6.3.2.5.1	Display.....	40
6.3.2.5.2	Execute.....	40
6.3.3	AV Player	41
6.3.3.1	Overview	41
6.3.3.1.1	Block Diagram	41
6.3.3.1.2	Flow	42
6.3.3.2	Functions.....	42
6.3.3.2.1	Feature	42
6.3.3.2.2	Required packages	42
6.3.3.3	GUI Guide	43
6.3.3.3.1	Audio Play.....	43
6.3.3.3.2	Video Play.....	45
7	Reference.....	46
8	Revision History.....	47
8.1	Rev. 2.10: 2018-09-05.....	47
8.2	Rev. 2.01: 2018-06-29.....	47
8.3	Rev. 2.00: 2018-05-31.....	47
8.4	Rev. 1.08: 2017-09-19.....	47
8.5	Rev. 1.07: 2016-09-07.....	47
8.6	Rev. 1.06: 2016-08-10.....	47
8.7	Rev. 1.05: 2016-04-26.....	47
8.8	Rev. 1.04: 2015-11-30.....	47
8.9	Rev. 1.03: 2015-09-16.....	47
8.10	Rev. 1.02: 2015-08-31.....	47
8.11	Rev. 1.01: 2015-05-27.....	47
8.12	Rev. 1.00: 2015-03-31.....	47

Figures

Figure 2.1	TCC8030 Board Description	6
Figure 3.1	Yocto Project Development Process	9
Figure 4.1	VTC Driver Installation Screen.....	24
Figure 4.2	Adding FWDN Bootloader	24
Figure 4.3	Adding Micom Rom	25
Figure 4.4	FWDN Bootloader Image Load	25

Figure 4.5 FWDN Area Map	26
Figure 4.6 SD Data.....	27
Figure 4.7 Set Number of Partition of SD Data	27
Figure 4.8 Set Partition Mode of SD Data to Use GPT.....	27
Figure 4.9 Setup Partition 1.....	28
Figure 4.10 Setup Partition 2.....	28
Figure 4.11 Setup Partition 5.....	28
Figure 4.12 Setup Partition 3.....	28
Figure 4.13 Setup Partition 4.....	28
Figure 4.14 Setup Partition 6.....	29
Figure 4.15 Setup Partition 7.....	29
Figure 4.16 Setup Partition 8.....	29
Figure 4.17 Setup Partition 9.....	29
Figure 4.18 Setup Partition 10.....	30
Figure 4.19 Making the FAI File	30
Figure 4.20 Automatically add the FAI File	30
Figure 4.21 Writing ALS Firmware.....	31
Figure 4.22 Complete write ALS Firmware.....	31
Figure 4.23 Firmware Write Option	32
Figure 4.24 Bootloader Image Update	32
Figure 4.25 Configuring Partition for FAI Update.....	33
Figure 5.1 Run fastboot.....	34
Figure 5.2 Connected fastboot.....	34
Figure 6.1 Conversion to Booting screen	36
Figure 6.2 Console Log in	36
Figure 6.3 ALS Architecture.....	37
Figure 6.4 Document range.....	39
Figure 6.5 Launcher block diagram.....	40
Figure 6.6 Launcher GUI.....	40
Figure 6.7 AV player Block Diagram.....	41
Figure 6.8 AV player Sequence diagram	42
Figure 6.9 Audio Play Screen.....	43
Figure 6.10 Audio Information Pop-up.....	43
Figure 6.11 Audio File List Screen.....	44
Figure 6.12 Audio Meta List Screen	45
Figure 6.13 Video Play Screen.....	45

Tables

Table 2.1 Set to USB Mode	7
Table 2.2 Set to SNOR Boot Mode.....	7
Table 3.1 Required Packages for the Host Development System	10
Table 4.1 Software Requirement.....	22
Table 4.2 Hardware Requirement.....	22
Table 4.3 ALS Firmware Construction	23
Table 4.4 SD Data Partition Information	30
Table 5.1 TCC8030 Partition	35
Table 5.2 Partition write example	35

1 INTRODUCTION

This document provides an explanation of ALS (Automotive Linux SDK) usage as followings:

- Board Description (TCC803x EVB)
- Build Guide for ALS
- FWDN Guide for ALS
- Guide of booting on TCC803x EVB and running application on ALS

2 BOARD DESCRIPTION

2.1 TCC803x EVB

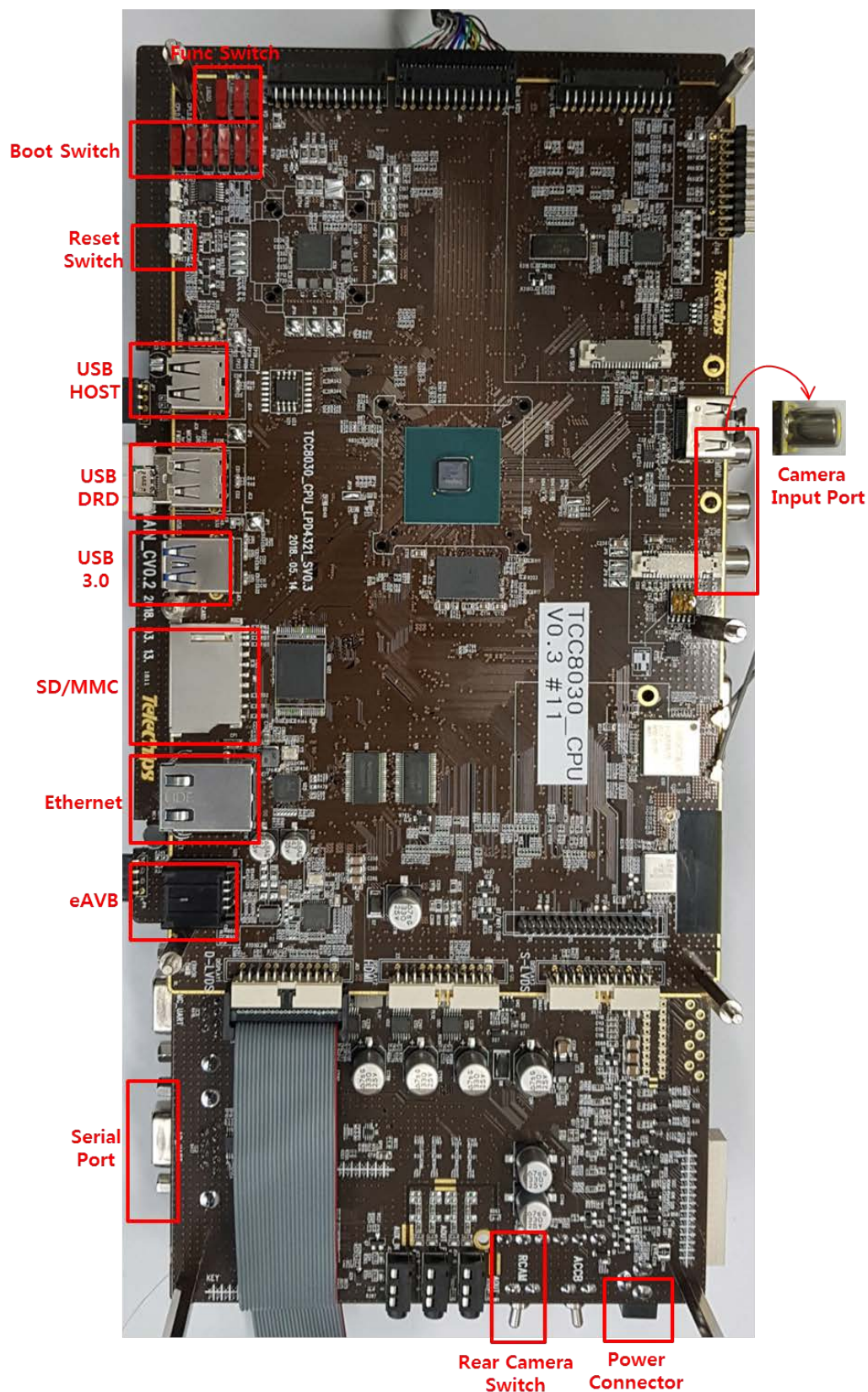

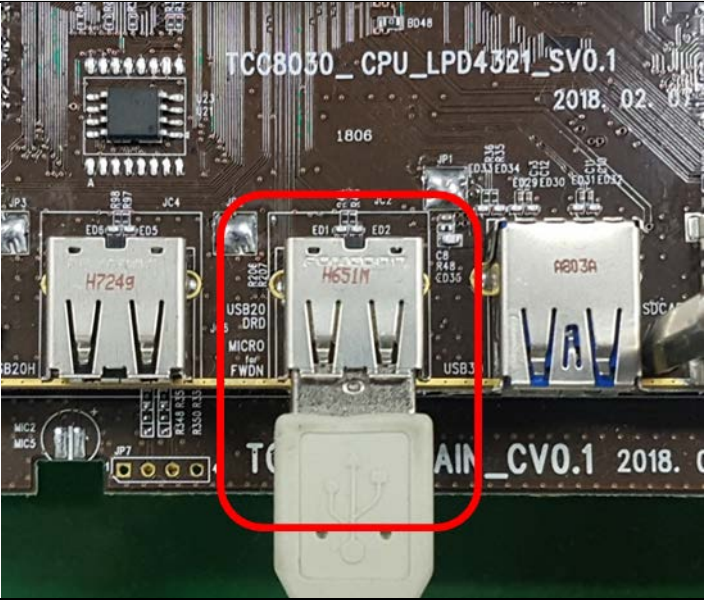

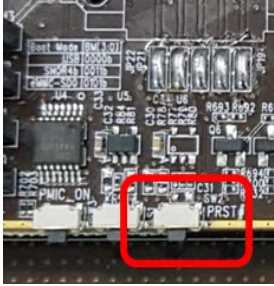


Figure 2.1 TCC803x Board Description

2.1.1 USB Mode Switch

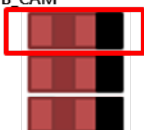

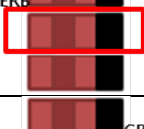
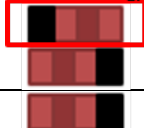


Table 2.1 Set to USB Mode		
Category	USB Mode Switch	FWDN Port
TCC803x	<div><div><div>H</div><div>L</div></div><div></div><div>USB Boot</div></div>	

2.1.2 SNOR Boot Mode Switch

Table 2.2 Set to SNOR Boot Mode			
Category	SNOR Boot Mode Switch	Reset Switch	Note
TCC803x	<div><div><div>H</div><div>L</div></div><div></div><div>SNOR 4B</div></div>		Reset button on the bottom left corner of the board.

2.1.3 Function Switch

Table 2.3 Set to Functions

Category	Function Switch	Description
Camera		Use external camera port
		Use internal camera port
Tuner / USB		Use Tuner B (Broadcasting control)
		Use USB (Enable USB power control)
Network		Use e-AVB Port
		Use Ethernet Port

Note: Refer to “TCC803x Automotive Common Hardware-User Guide for EVB” and “TCC803x Automotive Common Hardware-Quick Start Guide for EVB” for more details.

3 BUILD GUIDE

3.1 SDK Build Preparation

The Automotive Linux SDK (ALS) is based on the Yocto Project 2.4 Rocko. Therefore, the Yocto Project environment must be set on the host PC to use the ALS.

3.2 Yocto Project

The Yocto Project is an open source project which focuses on embedded Linux developer. And it uses Poky which is a combination of OpenEmbedded project and bitbake as build system to make Linux Images. Using Yocto Project, the bootloader, kernel, and rootfs can be built all at once.

3.3 Task Process

The task procedure of Yocto Project is described in Figure 3.1. A user can download source from upstream based on metadata and can perform build. Once build is completed, package, image, and SDK are provided as results.

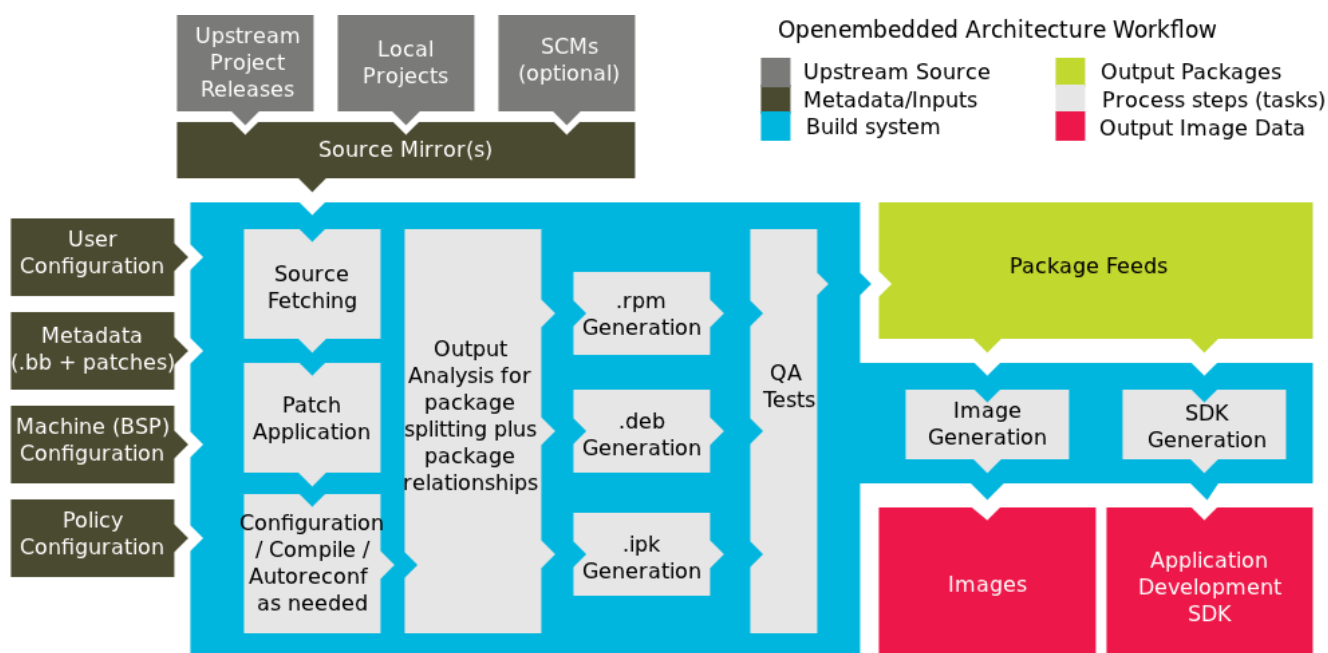


Figure 3.1 Yocto Project Development Process

3.4 Yocto Project Installation

3.4.1 Configuration for Yocto Project

Following mandatory packages must be installed on Host System (individual pc or development server) in order to use Yocto Project.

For details, refer to the Yocto Project website below;

<https://www.yoctoproject.org/docs/2.4.3/yocto-project-gs/yocto-project-gs.html>

3.4.2 Linux distribution versions supported by Yocto Project

Following Linux distribution versions are supported. Other distribution versions have not passed the verification process at Yocto project.

For details, refer to the Yocto Project website below;

<https://www.yoctoproject.org/docs/2.4.3/ref-manual/ref-manual.html#detailed-supported-distros>

- Ubuntu 14.10
- Ubuntu 15.04
- Ubuntu 15.10
- Ubuntu 16.04 (LTS)
- Fedora release 22
- Fedora release 23
- Fedora release 24
- CentOS release 7.x
- Debian GNU/Linux 8.x (Jessie)
- Debian GNU/Linux 9.x (Stretch)
- openSUSE 13.2
- openSUSE 42.1

3.4.3 List of package required for using Yocto Project

Following packages must be installed to use Yocto Project.

For details, refer to the Yocto Project website below;

<https://www.yoctoproject.org/docs/2.4.3/ref-manual/ref-manual.html#required-packages-for-the-host-development-system>

Table 3.1 Required Packages for the Host Development System

Linux Distribution	Need Packages
Ubuntu and Debian	\$ sudo apt-get install gawk wget git-core diffstat unzip texinfo gcc-multilib \ build-essential chrpath socat cpio python python3 python3-pip python3-pexpect \ xz-utils debianutils iputils-ping libSDL1.2-dev xterm
Fedora	\$ sudo dnf install gawk make wget tar bzip2 gzip python3 unzip perl patch \ diffutils diffstat git cpp gcc gcc-c++ glibc-devel texinfo chrpath ccache \ perl-Data-Dumper perl-Text-ParseWords perl-Thread-Queue perl-bignum \ socat python3-pexpect findutils which file cpio python python3-pip xz \ SDL-devel xterm
openSUSE	\$ sudo zypper install python gcc gcc-c++ git chrpath make wget python-xml \ diffstat makeinfo python-curses patch socat python3 python3-curses tar python3-pip \ python3-pexpect xz which libSDL-devel xterm
CentOS	\$ sudo yum install -y epel-release \$ sudo yum makecache \$ sudo yum install gawk make wget tar bzip2 gzip python unzip perl patch \ diffutils diffstat git cpp gcc gcc-c++ glibc-devel texinfo chrpath socat \ perl-Data-Dumper perl-Text-ParseWords perl-Thread-Queue python3-pip \ xz which SDL-devel xterm

3.5 ALS Installation

The ALS can be downloaded through android repo.

3.5.1 repo Installation

Refer to the website below to install repo.

<https://source.android.com/source/downloading.html>

If repo is already installed, you can use it without re-installation.

3.5.2 Download ALS

You can download the ALS using repo and script as follows.

```
~/works/Yocto/release/ALS$ repo init -u ssh://git.telechips.com/linux_ivi/manifest.git -m als_v3.0.2.xml
repo has been initialized in /home/user/works/Yocto/release
~/works/Yocto/release/ALS$ repo sync
~/works/Yocto/release/ALS$ poky/download.sh
This may take a long time depending on your network environment.
Continue? (Y/n) => Y
Create source-mirror directory
Choose ALS version
  1. als_v3.0.0
  2. als_v3.0.1
  3. als_v3.0.2
  4. subcore
select number(1-4) => 3
Create tools directory
Start tools downloading...done
Create source-mirror directory
Start source mirror downloading...done
~/works/Yocto/release/ALS$
```

Note: To use “download.sh”, ncftp must be installed on your PC.

3.5.3 Composition of ALS

Once download is completed, following items can be checked.

Item		Descriptions
Document		Documents of Automotive Linux SDK
poky	meta	Yocto Project 2.4 Rocko build system
	meta-poky	
	meta-yocto-bsp	
	meta-qt5	Support Qt5 5.6.3 Layer
	meta-linaro	Support Linaro toolchain Layer
	meta-telechips	meta-bsp
		meta-core
		meta-ivi
	als-patch	Telechips common patches
download.sh		Script for downloading Source-mirror & tools & FWDN
source-mirror		Local repository for building ALS basic class
tools		Yocto Project 2.4 Rocko buildtools (x86_64-buildtools-nativesdk-standalone-2.4.3.sh) Application Development Toolchain (ADT)
FWDN		FWDN execute file VTC driver

3.5.4 Yocto Project buildtools Installation

Install buildtools to configure the build environment of Yocto Project for the ALS as follows:

After installation is completed, you can check the following files.

Installation should be done one time at starting.

```
~/works/Yocto/release/ALS$ tools/x86_64-buildtools-nativesdk-standalone-2.4.3.sh
```

```
Build tools installer version 2.4.3
```

```
=====
```

```
Enter target directory for SDK (default: /opt/poky/2.4.3): ~/works/Yocto/release/ALS/buildtools
```

```
You are about to install the SDK to "/export/home4/B110141/thshin/Yocto/release/ALS/buildtools". Proceed[Y/n]? Y
```

```
Extracting SDK.....done
```

```
Setting it up...done
```

```
SDK has been successfully set up and is ready to be used.
```

```
Each time you wish to use the SDK in a new shell session, you need to source the environment setup script e.g.
```

```
$ . /export/home4/B110141/thshin/Yocto/release/ALS/buildtools/environment-setup-x86_64-pokysdk-linux
```

```
~/works/Yocto/release/ALS$ ls buildtools/
```

```
environment-setup-x86_64-pokysdk-linux  sysroots/  version-x86_64-pokysdk-linux
```

```
~/works/Yocto/release/ALS$
```

3.6 Building ALS

Execute following process to build the ALS provided.

When you switch to the build environment, several environment variables such as PATH will change, so some commands that are not related to the build may cause malfunction. In that case, proceed in a new shell.

3.6.1 Configuring the Yocto Project build environment

Before proceeding with the build, change to the Yocto Project build environment first.

```
~/works/Yocto/release/ALS$ source buildtools/environment-setup-x86_64-pokysdk-linux
~/works/Yocto/release/ALS$
```

3.6.2 Bitbake Configuration

Yocto Project builds with bitbake. Therefore, you need to change to the bitbake environment.

There are two ways to set environment: Auto script and User definition.

3.6.2.1 Configuration using Auto script

Use the als-build.sh script to use the default settings.

If you select machine as below, it automatically changes to bitbake environment.

```
~/works/Yocto/release/ALS$ source poky/als-build.sh
Choose MACHINE
  1. tcc8030
  2. tcc8031
select number(1-2) => 2 machine(tcc8031) selected.
You had no conf/local.conf file. This configuration file has therefore been created for you with some default values. You may
wish to edit it to, for example, select a different MACHINE (target hardware). See conf/local.conf for more information as
common configuration options are commented.

You had no conf/bblayers.conf file. This configuration file has therefore been created for you with some default values. To add
additional metadata layers into your configuration please add entries to conf/bblayers.conf.

The Yocto Project has extensive documentation about OE including a reference manual which can be found at:
http://yoctoproject.org/documentation

For more information about OpenEmbedded see their website:
http://www.openembedded.org/

Yocto Project common targets are:
  core-image-minimal
  core-image-sato
  meta-toolchain
  adt-installer
  meta-ide-support

Telechips common targets are:
  telechips-ivi-image-minimal
  telechips-ivi-image-gstreamer(minimal + GStreamer)
  telechips-ivi-image-qt(minimal + Qt)
  telechips-ivi-image(minimal + GStreamer + Qt)

  meta-toolchain-telechips(Application Development Toolkit)
  meta-toolchain-telechips-ivi(Application Development Toolkit include GStreamer)
  meta-toolchain-telechips-qt5(Application Development Toolkit include GStreamer and Qt5)

Telechips Automotive Linux Platform targets are:
  automotive-linux-platform-image(telechips-ivi-image + demo applications)

You can also run generated qemu images with a command like 'runqemu qemux86'
or
You can also run generated Telechips images on Telechips EVB Boards(e.g. tcc8030)

~/works/linux-avn/release/ALS/build/tcc8031$
```

3.6.3 User definition Configuration

For user definition settings, select the TEMPLATECONF environment variable, build directory, machine, and so on as follows:

3.6.3.1 Setting Environmental Variable of Template

Register environmental variable in order to use template provided by Telechips.
Use an absolute path below.

```
~/works/Yocto/release/ALS$ export TEMPLATECONF=~/works/Yocto/release/ALS/poky/meta-telechips/template
```

3.6.3.2 Bitbake Configuration

Change to Bitbake build environment.

In "source poky/oe-init-build-env build/tcc8030", tcc8030 is the directory where the build proceeds.

This directory is automatically generated according to the settings you have made.

You can ignore Chapter 3.6.3.1 if the build directory was previously used.

```
~/works/Yocto/release/ALS$ mkdir build
```

```
~/works/Yocto/release/ALS$ source poky/oe-init-build-env build/tcc8030
```

Yocto Project common targets are:

- core-image-minimal
- core-image-sato
- meta-toolchain
- adt-installer
- meta-ide-support

Telechips common targets are:

- telechips-ivi-image-minimal
- telechips-ivi-image-gstreamer(minimal + GStreamer)
- telechips-ivi-image-qt(minimal + Qt)
- telechips-ivi-image(minimal + GStreamer + Qt)

- meta-toolchain-telechips(Application Development Toolkit)
- meta-toolchain-telechips-ivi(Application Development Toolkit include GStreamer)
- meta-toolchain-telechips-qt5(Application Development Toolkit include GStreamer and Qt5)

Telechips Automotive Linux Platform targets are:

- automotive-linux-platform-image(telechips-ivi-image + demo applications)

You can also run generated qemu images with a command like 'runqemu qemux86'

or

You can also run generated Telechips images on Telechips EVB (e.g. tcc8030)

```
~/works/Yocto/release/ALS/build/tcc8030$
```

3.6.3.3 Setup Configuration

Set conf/local.conf as shown below according to the target board.

Name of variable	Default value	Description
BB_NUMBER_THREADS	8	Maximum number of tasks that can be run simultaneously
PARALLEL_MAKE	-j 16	Option for make
MACHINE	N/A	target board setting
EXTRA_IMAGE_FEATURES	debug-tweaks read-only-rootfs	Image property selected additionally. For detailed options, refer to conf/local.conf
CORE_IMAGE_EXTRA_INSTALL	NULL	Name of package to be installed additionally on image
MY_MAC_ADDRESS	FF:FF:FF:FF:FF:FF	Ethernet hardware address of target board
MY_IP_ADDRESS	192.168.1.1	TCP/IP IP address of target board
MY_GATEWAY_ADDRESS	192.168.1.254	TCP/IP default gateway address of target board
DISTRO	poky-telechips-systemd	Select SDK type poky-telechips (sysvinit) poky-telechips-systemd (systemd)
INCOMPATIBLE_LICENSE	GPLv3, GPL-3.0, LGPLv3, LGPL-3.0, AGPLv3, AGPL-3.0	License type to be excluded from SDK

3.6.3.4 Setup Graphics System and Qt Platform Abstraction (QPA)

Use INVITE_PLATFORM variable at conf/local.conf file in order to set linux graphics system and QPA.
Default value is Qt5/Wayland.

Graphics System and QPA	Configuration
Qt5/Wayland	INVITE_PLATFORM = "qt5" INVITE_PLATFORM += "qt5/wayland" INVITE_PLATFORM += "drm" DISTRO_FEATURES_append = " wayland" DISTRO_FEATURES_append = " opengl"
Qt5/EGL FS	INVITE_PLATFORM = "qt5" INVITE_PLATFORM += "qt5/eglfs" DISTRO_FEATURES_append = " opengl"

3.6.4 Build automotive-linux-platform-image

Use bitbake to build image as shown below.

3.6.4.1 Build image type

The build image provided by ALS is as follows.

3.6.4.1.1 telechips-ivi-image-minimal

Only the minimum packages (dbus, alsa, ...) that make up ALS are installed.

3.6.4.1.2 telechips-ivi-image-gstreamer

The gstreamer package is added to telechips-ivi-image-minimal.

3.6.4.1.3 telechips-ivi-image

The graphic manager and Qt platform are added to telechips-ivi-image-gstreamer.

3.6.4.1.4 automotive-linux-platform-image

The demo application is added to telechips-ivi-image.

```
~/works/Yocto/release/ALS/build/tcc8031$ bitbake automotive-linux-platform-image
Loading cache: 100% |#####| Time: 0:00:00
Loaded 1803 entries from dependency cache.
Parsing recipes: 100% |#####| Time: 0:00:01
Parsing of 1257 .bb files complete (1249 cached, 8 parsed). 1813 targets, 277 skipped, 0 masked, 0 errors.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION           = "1.36.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING      = "universal-4.8"
TARGET_SYS           = "arm-telechips-linux-gnueabi"
MACHINE              = "tcc8031"
DISTRO               = "poky-telechips-systemd"
DISTRO_VERSION        = "2.4.3"
TUNE_FEATURES        = "arm aarch32 neon fp-armv8 callconvention-hard cortexa53"
TARGET_FPU           = "hard"
KBUILD_DEFCONFIG      = "tcc803x_linux_avn_reduce_defconfig"
TCMODE               = "default"
INVITE_PLATFORM      = "qt5 qt5/wayland telechips-egl drm genivi micom fastboot "
IMAGE_FEATURES        = " debug-tweaks read-only-rootfs"
SDKIMAGE_FEATURES    = "dev-pkgs"
GCCVERSION           = "linaro-7.2"
GLIBCVERSION         = "linaro-2.20"
meta
meta-poky             = "v3.x:928c88b446056728acd65f20a5b8b99142a1aa37"
meta-linaro-toolchain = "v3.x:5d0509563bb17775a9cb6541069cf7a75473e6cf"
meta-bsp             = "v3.x:af53c9bba3f99ab702e5819dcdcf2f30599e5dd4"
meta-qt5             = "v3.x:3dec11dd621999d19cf4f5bbc2e7013b01be7dd3"
meta-core            = "v3.x:af53c9bba3f99ab702e5819dcdcf2f30599e5dd4"
meta-qt5             = "v3.x:af53c9bba3f99ab702e5819dcdcf2f30599e5dd4"
meta-ivi             = "v3.x:3023dde792730f841ea65606b63f2ba1cf98faf4"
meta-genivi          = "v3.x:be26d8ddba205b343993ca5c2017f8dba45ea49e"
meta-ivi             = "v3.x:be26d8ddba205b343993ca5c2017f8dba45ea49e"

Initialising tasks: 100% |#####| Time: 0:00:14
NOTE: Executing SetScene Tasks
NOTE: Executing RunQueue Tasks
NOTE: Tasks Summary: Attempted 4704 tasks of which 4099 didn't need to be rerun and all succeeded.
NOTE: Writing buildhistory

Summary: There were 7 WARNING messages shown.
~/works/Yocto/release/ALS/build/tcc8031$
```

3.6.4.2 Location of results

Once build is completed successfully, the result comes in the following locations.

```
~/works/Yocto/release/ALS/build/tcc8031$ ls -l tmp/deploy/images/tcc8031/  
automotive-linux-platform-image-tcc8030.ext4  
zImage-tcc8030-linux-lpd4321_sv0.1.dtb  
initramfs-telechips-image-tcc8030.cpio.gz  
modules-tcc8030.tgz  
tc-boot-tcc8030.img  
tc-boot-tcc8030_uncompressed.img  
tcc803x_fwdn-tcc8030.rom  
micom_tcc8030.rom  
~/works/Yocto/release/ALS/build/tcc8030$
```

3.7 FAQs on Build

This chapter describes the frequently asked questions in Yocto Project environment.

3.7.1 How to rebuild package

If you want to rebuild package, you can clean and build as bellow.
There are three types of clean tasks:

3.7.1.1 clean and build

Deletes all output (after task do_unpack). However, shared state cache data is not cleared.

```
~/works/Yocto/release/ALS/build/tcc8030$ bitbake alsa-lib -c clean
~/works/Yocto/release/ALS/build/tcc8030$ bitbake alsa-lib
```

3.7.1.2 cleanall and build

It clears all data including all results, shared state cache data, and downloaded source code.

```
~/works/Yocto/release/ALS/build/tcc8030$ bitbake alsa-lib -c cleanall
~/works/Yocto/release/ALS/build/tcc8030$ bitbake alsa-lib
```

3.7.1.3 cleansstate and build

Clears all output and shared state cache data.

```
~/works/Yocto/release/ALS/build/tcc8030$ bitbake alsa-lib -c cleansstate
~/works/Yocto/release/ALS/build/tcc8030$ bitbake alsa-lib
```

3.7.2 How to rebuild all packages

Yocto Project provides single-build type per one recipe, so does not have total rebuild function. If you want to rebuild all packages, you can clean each package and build image as bellow.

```
~/works/Yocto/release/ALS/build/tcc8030$ bitbake -c cleanall alsa-lib
~/works/Yocto/release/ALS/build/tcc8030$ bitbake -c cleanall dbus
~/works/Yocto/release/ALS/build/tcc8030$ bitbake -c cleanall gstreamer1.0
~/works/Yocto/release/ALS/build/tcc8030$ bitbake automotive-linux-platform-image -c cleanall
~/works/Yocto/release/ALS/build/tcc8030$ bitbake automotive-linux-platform-image
```

3.7.3 How to modify source code

The package's source located in `$(TMPDIR)/work/$(MULTIMACH_TARGET_SYS)/$(PN)/$(EXTENDPE)$(PV)-${PR}/git` or unpacked directory name.

To modify and apply the source code, you must proceed with the force option as follows:

```
~/works/Yocto/release/ALS/build/tcc8030$ pushd tmp/work/arm-telechips-linux-gnueabi/tc-launcher/1.0.0-r0/git
modify source codes
~/works/Yocto/release/ALS/build/tcc8030/ tmp/work/arm-telechips-linux-gnueabi/tc-launcher/1.0.0-r0/git$ popd
~/works/Yocto/release/ALS/build/tcc8030$ bitbake tc-launcher -f -c compile
~/works/Yocto/release/ALS/build/tcc8030$ bitbake tc-launcher
```

3.7.4 How to add extra package to image

If you want to add package to image, refer to below;

You can set up the `CORE_IMAGE_EXTRA_INSTALL` variable in `conf/local.conf` to install additional packages to the image you are currently building.

- `CORE_IMAGE_EXTRA_INSTALL += "openssh"`

3.7.5 How to modify rootfs image without modify recipes

If you temporarily modify rootfs and create an image, run `do_image` with the force option as follows:

```
~/works/Yocto/release/ALS/build/tcc8030$ pushd tmp/work/tcc8030-telechips-linux/automotive-linux-platform-image/1.0-r0/rootfs/
modify rootfs
~/works/Yocto/release/ALS/build/tcc8030/ tmp/work/tcc8030-telechips-linux/automotive-linux-platform-image/1.0-r0/rootfs/ $ popd
~/works/Yocto/release/ALS/build/tcc8030$ bitbake automotive-linux-platform-image -f -c image
~/works/Yocto/release/ALS/build/tcc8030$ bitbake automotive-linux-platform-image
```

3.7.6 How to change from read-only to writable rootfs

The default setting of ALS rootfs is read-only. If you want to use writable rootfs, you can change IMAGE_FEATURES from conf/local.conf.

- read only: EXTRA_IMAGE_FEATURES = "debug-tweaks read-only-rootfs"
- writable: EXTRA_IMAGE_FEATURES = "debug-tweaks"

3.7.7 How to enable network environment

To use network in ALS, set conf / local.conf as follows.

- activate kernel config and systemd network configuration: USE_NETWORK = "1"
- MAC Address configuration: MY_MAC_ADDRESS = "F4:50:EB:DF:5D:6F"
- IP Address configuration: MY_IP_ADDRESS = "192.168.21.85"
- Gateway Address configuration: MY_GATEWAY_ADDRESS = "192.168.21.254"

3.7.8 How to enable ssh environment

ALS supports ssh server based on openssh. OpenSSH is installed by adding ssh server to EXTRA_IMAGE_FEATURES as follows.

- EXTRA_IMAGE_FEATURES += "ssh-server-openssh"

USE_NETWORK in Chapter 3.7.7 is automatically activated when setting ssh server. However, IP address, MAC address, and Gateway address must be set according to the user's environment.

3.7.9 How to install gdb environment

To use gdb with ALS, change the EXTRA_IMAGE_FEATURES and INCOMPATIBLE_LICENSE settings as follows:

- Installing GDB-related packages: EXTRA_IMAGE_FEATURES = "debug-tweaks tools-debug dbg-pkgs"
- Disable incompatible setting: #INCOMPATIBLE_LICENSE = "GPLv3 GPL-3.0 LGPLv3 LGPL-3.0 AGPLv3 AGPL-3.0"

3.8 Building Application Development Toolkit (ADT)

3.8.1 Bitbake Configuration

Change to bitbake build environment as in Chapter 3.6.2. If this is already done, this step is optional.

3.8.2 ADT type provided by ALS

ADT in ALS v2.5 supports three types as below;

3.8.2.1 meta-toolchain-telechips (Application Development Toolkit)

Packages for building basic ALS-based programs are installed.

For detailed installation packages, refer to the following file.

- meta-telechips/meta-core/recipes-core/packagegroups/packagegroup-als-toolchain-target.bb

3.8.2.2 meta-toolchain-telechips-ivi (Application Development Toolkit include GStreamer)

Additional packages for building gstreamer and ALS demo applications in meta-toolchain-telechips are installed.

For detailed installation packages, refer to the following file.

- meta-telechips/meta-ivi/recipes-core/packagegroups/packagegroup-als-ivi-toolchain-target.bb

3.8.2.3 meta-toolchain-telechips-qt5 (Application Development Toolkit include GStreamer and Qt5)

Additional Qt5 packages are installed in meta-toolchain-telechips-ivi.

For detailed installation packages, refer to the following file.

- meta-qt5/recipes-qt/packagegroups/packagegroup-qt5-toolchain-target.bb

3.8.3 Build Application Development Toolkit (ADT)

To build the ADT, use the bitbake as follows:

```
~/works/Yocto/release/ALS/build/tcc8031$ bitbake meta-toolchain-telechips-qt5
```

```
Loading cache: 100% |#####| Time: 0:00:00
Loaded 1794 entries from dependency cache.
```

```
Parsing recipes: 100% |#####| Time: 0:00:01
```

```
Parsing of 1248 .bb files complete (1240 cached, 8 parsed). 1804 targets, 277 skipped, 0 masked, 0 errors.
```

```
NOTE: Resolving any missing task queue dependencies
```

Build Configuration:

```
BB_VERSION           = "1.36.0"
BUILD_SYS            = "x86_64-linux"
NATIVELSBSTRING     = "universal-4.8"
TARGET_SYS          = "arm-telechips-linux-gnueabi"
MACHINE              = "tcc8031"
DISTRO               = "poky-telechips-systemd"
DISTRO_VERSION       = "2.4.3"
TUNE_FEATURES        = " arm aarch32 neon fp-armv8 callconvention-hard cortexa53"
TARGET_FPU           = "hard"
KBUILD_DEFCONFIG      = "tcc803x_linux_avn_reduce_defconfig"
TCMODE               = "default"
INVITE_PLATFORM      = "qt5 qt5/wayland telechips-egl drm genivi micom fastboot "
IMAGE_FEATURES       = " debug-tweaks read-only-rootfs"
SDKIMAGE_FEATURES    = "dev-pkgs"
GCCVERSION           = "linaro-7.2"
GLIBCVERSION         = "linaro-2.20"
```

meta

```
meta-poky            = "v3.x:a51bc0490cd80029985fe9ba36cec4911b1d74c0"
```

```
meta-linaro-toolchain = "v3.x:5d0509563bb17775a9cb6541069cf7a75473e6cf"
```

```
meta-bsp             = "v3.x:af53c9bba3f99ab702e5819dcdcf2f30599e5dd4"
```

```
meta-qt5             = "v3.x:3dec11dd621999d19cf4f5bbc2e7013b01be7dd3"
```

```
meta-core            = "v3.x:af53c9bba3f99ab702e5819dcdcf2f30599e5dd4"
```

```
meta-ivi             = "v3.x:be26d8ddba205b343993ca5c2017f8dba45ea49e"
```

```
meta-qt5             = "v3.x:af53c9bba3f99ab702e5819dcdcf2f30599e5dd4"
```

meta-ivi

```
meta-genivi          = "v3.x:3023dde792730f841ea65606b63f2ba1cf98faf4"
```

```
Initialising tasks: 100% |#####| Time: 0:00:19
```

```
NOTE: Executing SetScene Tasks
```

```
NOTE: Executing RunQueue Tasks
```

```
NOTE: Tasks Summary: Attempted 6329 tasks of which 6312 didn't need to be rerun and all succeeded.
```

```
NOTE: Writing buildhistory
```

```
~/works/Yocto/release/ALS/build/tcc8031$
```

3.8.4 Location of result

Once build is completed successfully, the results come in following locations.

```
~/works/Yocto/release/ALS/build/tcc8031$ ls -l tmp/deploy/sdk
```

```
telechips-als-v3.0.2-toolchain-cortexa53hf-neon-fp-armv8-opengl-wayland-ivi-qt5-x86_64-gcc-linaro-7.2.host.manifest
```

```
telechips-als-v3.0.2-toolchain-cortexa53hf-neon-fp-armv8-opengl-wayland-ivi-qt5-x86_64-gcc-linaro-7.2.sh
```

```
telechips-als-v3.0.2-toolchain-cortexa53hf-neon-fp-armv8-opengl-wayland-ivi-qt5-x86_64-gcc-linaro-7.2.target.manifest
```

```
telechips-als-v3.0.2-toolchain-cortexa53hf-neon-fp-armv8-opengl-wayland-ivi-qt5-x86_64-gcc-linaro-7.2.testdata.json
```

```
~/works/Yocto/release/ALS/build/tcc8031$
```

4 FWDN (FIRMWARE DOWNLOADER) GUIDE

This chapter provides how to download the ALS to 803x EVB and to login to the Linux console.

4.1 System Requirement

4.1.1 Software Requirement

Table 4.1 Software Requirement

Category	Description	Path	Note
VTC Driver	Window driver for PC used to download ALS Firmware. VTC driver corresponding to the Windows version must be installed. Install only once on your PC.	tools/vtcdrv/win.ver	You must use version 5.0.0.13 or higher. Refer to VTC Driver information file path: 'tools/vtcdrv/ReadMe/txt'
FWDN	Tools for downloading ALS Firmware.	tools/FWDN	You must use version 2.82 or higher.

4.1.2 Hardware Requirement

Table 4.2 Hardware Requirement

Category	Description	Note
Demo Board	Supported chipsets are TCC803x	
12V Power Adapter	Used to supply power to the board and LCD	Recommend using the power adapter (12V/5A) provided by Telechips.
USB Cable	Used to connect PC and Board	Refer to Chapter 2.1.1 for connection of USB cable.

4.2 ALS Firmware Construction

Table 4.3 ALS Firmware Construction

Image type	Description	Image location
U-Boot bootloader Image (* .rom)	An image to initialize the system hardware and put the Linux kernel image into memory for execution Refer to example below for the U-Boot image names used in each board Example: tcc80x_fwdn-[MACHINE].rom, tcc803x_fwdn-tcc8030.rom	Refer to Chapter 3.6.4.2
Kernel Image (* .img)	The Boot image file in which the Linux kernel is compressed. Refer to example below for the Kernel image names used in each board. Example: tc-boot-[MACHINE].img, tc-boot-tcc8030.img	Refer to Chapter 3.6.4.2
Root File System Image (* .ext4)	An image that serves as the root file system for Linux System. Refer to example below for the Root File System image names used in each board Example: automotive-linux-platform-image-[MACHINE].ext4, automotive-linux-platform-image-tcc8030.ext4	Refer to Chapter 3.6.4.2
RW Partition Image (Home directory) (* .ext4)	It is used to store the rw data of the database and application used in Media Player as the RW area required by ALS. Example: home-directory.ext4	Use Linux 'dd' command to create this file and specify where to create it. See the Note below.
DTB Image (Device Tree Binary) (* .dtb)	An Image that contains driver information about the hardware devices of each board. Refer to 'example' below for the dtb image names used in each board. Example: [MACHINE]-linux-lpd4321_sv0.1.dtb, zImage_tcc8030-linux-lpd4321_sv0.1.dtb	Refer to Chapter 3.6.4.2
Micom Image (* .rom)	It is Loader for waking up Micom. Refer to name below for the micom image Example:: micom_[MACHINE].rom, micom_tcc8030.rom)	Refer to Chapter 3.6.4.2

Note: Use Linux DD command to create RW file system for home directory image;

- \$ dd if=/dev/zero of=home-directory.ext4 bs=1K count=512000
- \$ mkfs.ext4 home-directory.ext4

4.3 Install VTC Driver

Install USB driver for FWDN to (VTC Driver ver. 5.0.0.13) Windows PC.

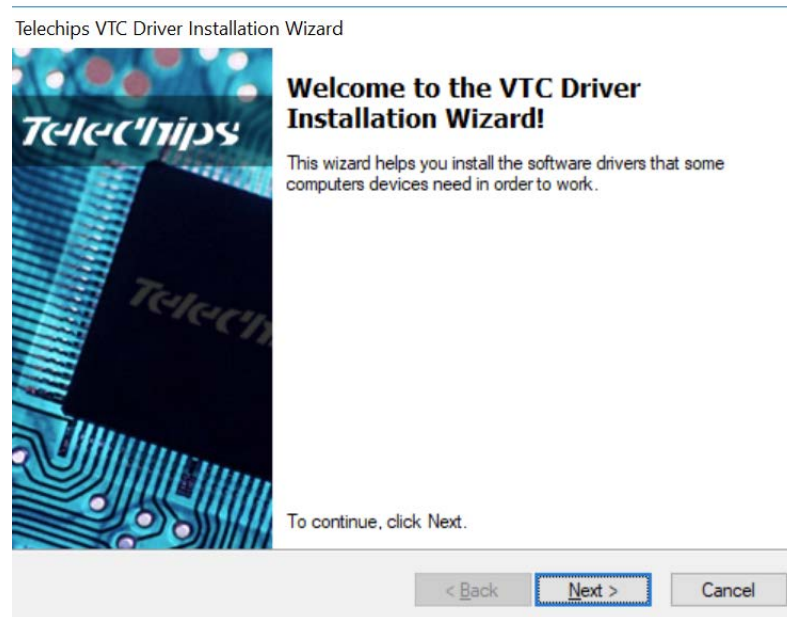


Figure 4.1 VTC Driver Installation Screen

This can be installed just once on PC.

4.4 ALS Firmware Download Sequence

Overall flow of FWDN is as follows;

- ① Connect Board and 12V Power adapter.
- ② Change Boot Switch to USB Mode.
- ③ Run FWDN software.
- ④ Add Boot loader (u-boot) to FWDN.
- ⑤ Add Micom Image to FWDN.
- ⑥ Connect USB cable to Board.
- ⑦ Constitute Partition.
- ⑧ Select Low format and click start button.
- ⑨ Change Boot switch to Boot mode and reset.

4.4.1 Set Board to USB Mode

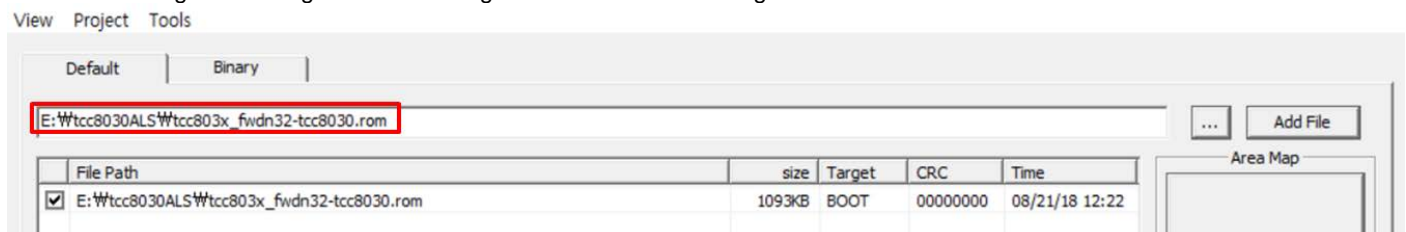
Refer to Chapter 2.1.1.

4.4.2 Set Board to SNOR Boot Mode

Refer to Chapter 2.1.2.

4.4.3 Add Bootloader (u-boot)

Find u-boot image for adding bootloader image to FWDN and Add Image.



- (1) U-boot Image Find
- (2) Image Load
- (3) Add Image

Figure 4.2 Adding FWDN Bootloader

4.4.4 Add Micom Image

Find Micom file and Add Image

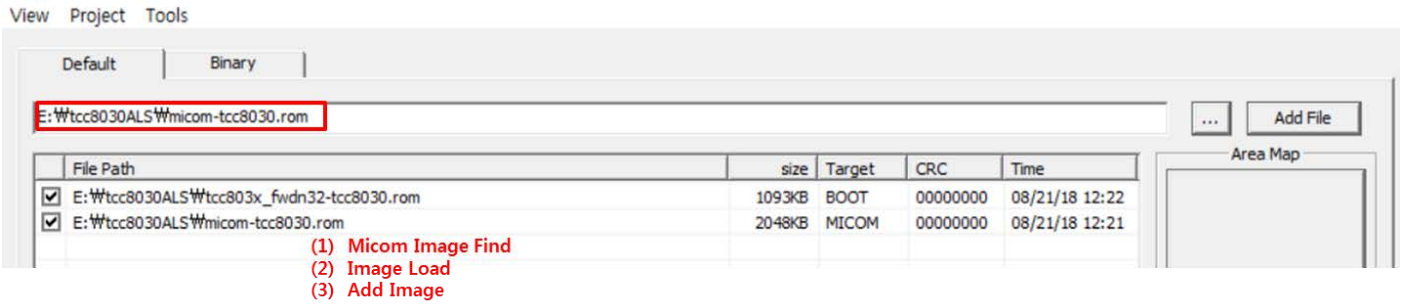


Figure 4.3 Adding Micom Rom

4.4.5 Connect Board and USB cable

If you connect board and USB cable (Refer to Chapter 2.1.1), Bootloader Image is loaded on board.

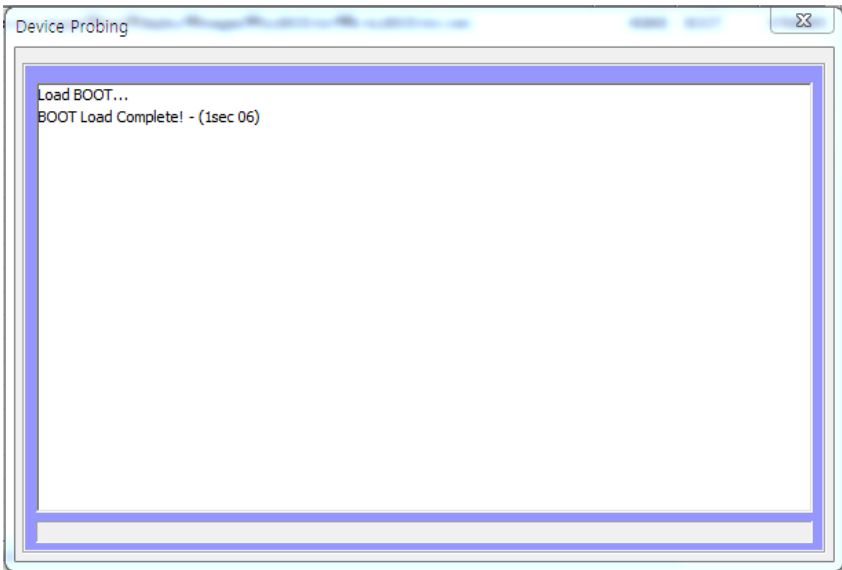


Figure 4.4 FWDN Bootloader Image Load

When the Bootloader is completed, check whether the Area Map of FWDN is enabled. The Area Map, which is activated according to the board type, is shown as below.

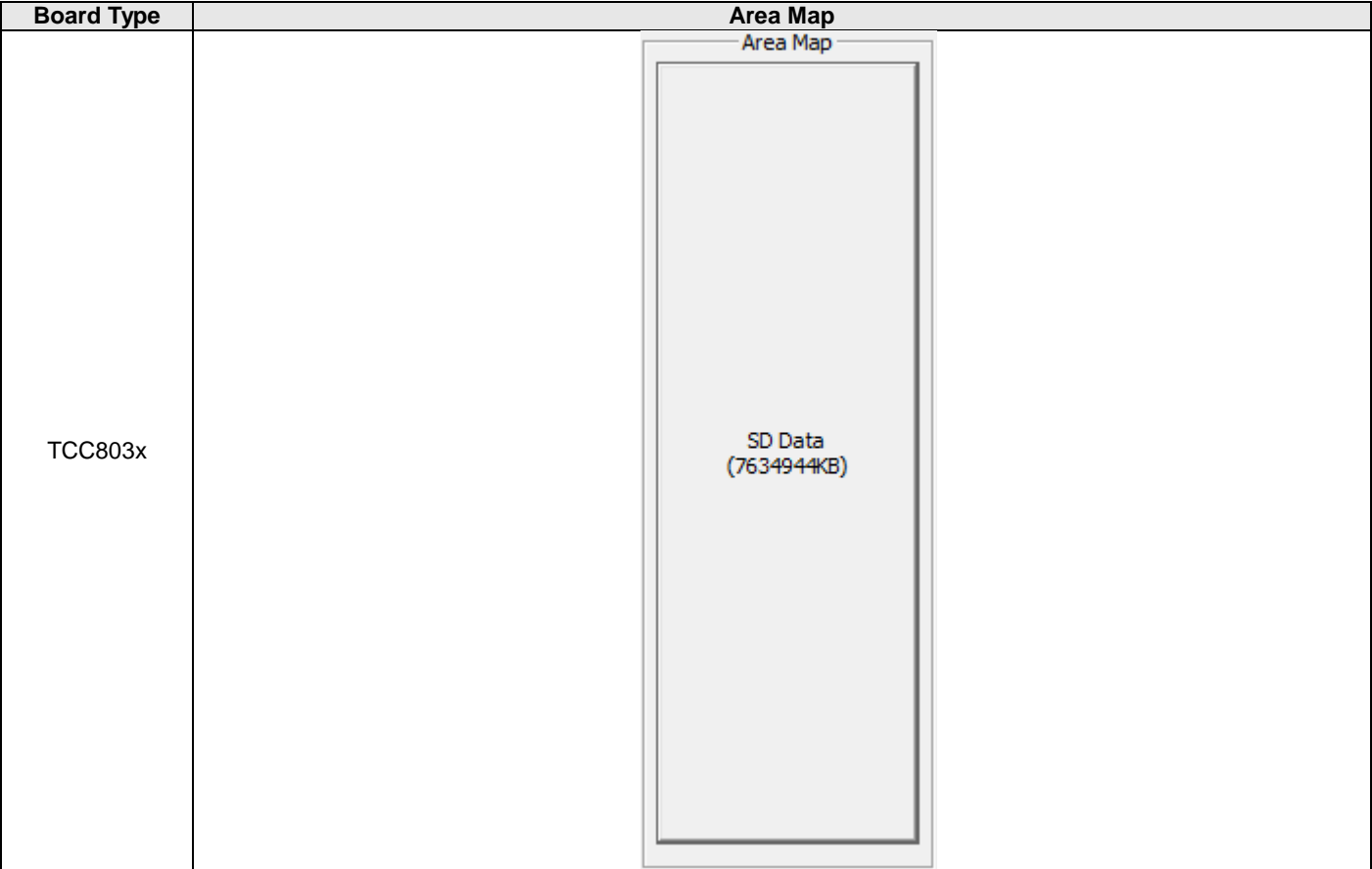


Figure 4.5 FWDN Area Map

4.4.6 Configuring Partition

This chapter describes the partition configuration of TCC803x.

4.4.6.1 Configuring Partition of SD Data

The configuring partition of SD Data is shown below.

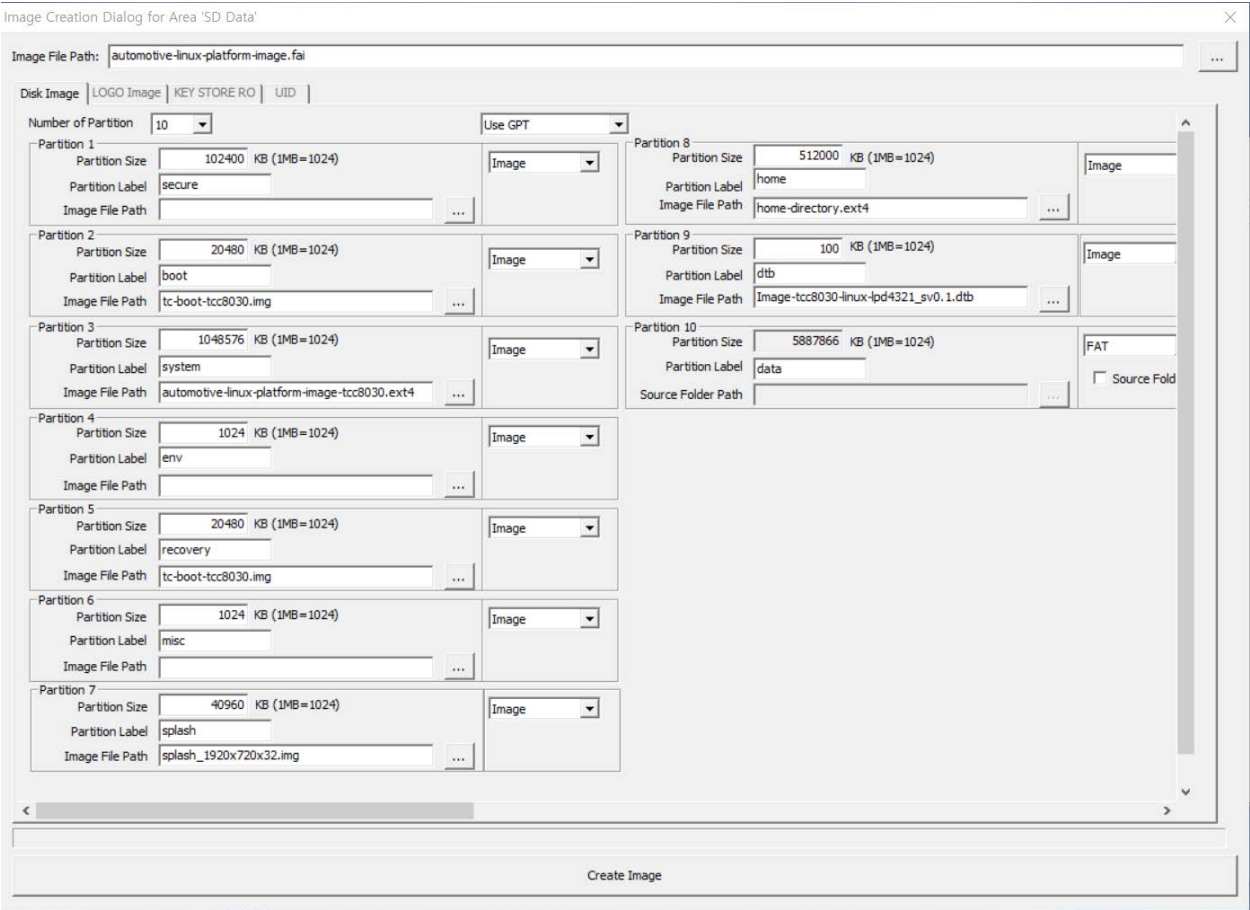


Figure 4.6 SD Data

The procedure for configuring partition of SD Data is as follows.

- ① Click SD Data button.
- ② Select '10' in Partition on the Image Creation Dialog for Area 'SD Data' screen.



Figure 4.7 Set Number of Partition of SD Data

- ③ Select **Use GPT** as the partition mode on the image Creation Dialog for Area 'SD Data' screen.

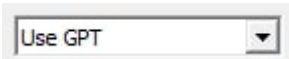


Figure 4.8 Set Partition Mode of SD Data to Use GPT

- ④ Configuring Partition
 - Setup Partition 1
 - (1) Enter '102400' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'secure' in the Label of Partition 1.

Partition 1		
Partition Size	102400 KB (1MB=1024)	Image
Partition Label	secure	
Image File Path		

Figure 4.9 Setup Partition 1

- Setup Partition 2, 5
 - (1) Enter '20480' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'boot' in the Label of Partition 2, enter 'recovery' in the Label of Partition 5.
 - (4) Click Find() button for adding Kernel Image File, and the selected file and path appear when you select a file that matches the board.

Note: In the figure below, the file path is deleted.

Partition 2		
Partition Size	20480 KB (1MB=1024)	Image
Partition Label	boot	
Image File Path	tc-boot-tcc8030.img	

Figure 4.10 Setup Partition 2

Partition 5		
Partition Size	20480 KB (1MB=1024)	Image
Partition Label	recovery	
Image File Path	tc-boot-tcc8030.img	

Figure 4.11 Setup Partition 5

- Setup Partition 3
 - (1) Enter '1048576' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'system' in the Label of Partition 3.
 - (4) Click Find() button for adding Root File System Image, and the selected file and path appear when you select a file that matches the board.

Note: In the figure below, the file path is deleted.

Partition 3		
Partition Size	1048576 KB (1MB=1024)	Image
Partition Label	system	
Image File Path	automotive-linux-platform-image-tcc8030.ext4	

Figure 4.12 Setup Partition 3

- Setup Partition 4
 - (1) Enter '1024' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'env' in the Label of Partition 4.

Partition 4		
Partition Size	1024 KB (1MB=1024)	Image
Partition Label	env	
Image File Path		

Figure 4.13 Setup Partition 4

- Setup Partition 6
 - (1) Enter '1024' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'misc' in the Label of Partition 6.

Figure 4.14 Setup Partition 6

- (Optional) Setup Partition 7
 - (1) Enter '40960' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'splash' in the label of Partition 7.
 - (4) Click Find(...) button for adding Parking Guide Line Image, and the selected file and path appear when you select a file that matches the board.

Note: In the figure below, the file path is deleted.

The 'splash.img' file is a parking line image used in EarlyCamera. If you are not using the EarlyCamera feature, leave it blank.

Note: EarlyCamera feature is not supported currently.

Figure 4.15 Setup Partition 7

- Setup Partition 8
 - (1) Enter '512000' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'home' in the Label of Partition 8.
 - (4) Click Find(...) button for adding RW Partition Image for Home directory, and the selected file and path appear when you select a file that matches the board.

Note: In the figure below, the file path is deleted.

Important: The Home Directory file format must be Linux Native, and it is recommended to use ext4.

Figure 4.16 Setup Partition 8

- Setup Partition 9
 - (1) Enter '100' for Partition Size.
 - (2) Select Image.
 - (3) Enter 'dtb' in the Label of Partition 9.
 - (4) Click Find(...) button for adding DTB File, and the selected file and path appear when you select a file that matches the board.

Note: In the figure below, the file path is deleted.

Figure 4.17 Setup Partition 9

- Setup Partition 10
 - (1) Select FAT.
 - (2) Enter 'data' in Partition Label.

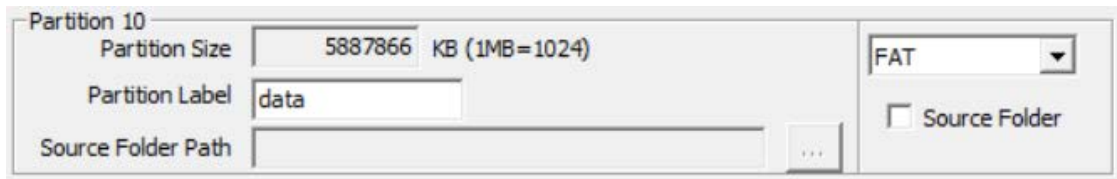



Figure 4.18 Setup Partition 10

⑤ Making a FAI file

To specify the FAI file path, click the Find() button in the **Image File Path** at the top of the *Image Creation Dialog for Area 'SD Data'* screen, and enter the name of the FAI file to be created and click the Save button. The FAI file to be created along with the file path appears.

Note: In the figure below, the file path is deleted.

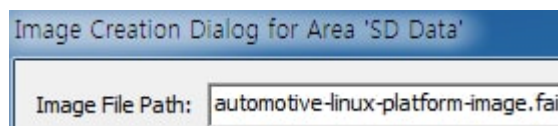


Figure 4.19 Making the FAI File

Click the **Create Image** button at the bottom of the '*Image Creation Dialog for Area SD Data*' screen.

When the FAI file is generated, the *Image Creation pop-up* appears. Clicking the **OK** button will automatically add the FAI file to the File Path of the FWDN tool.

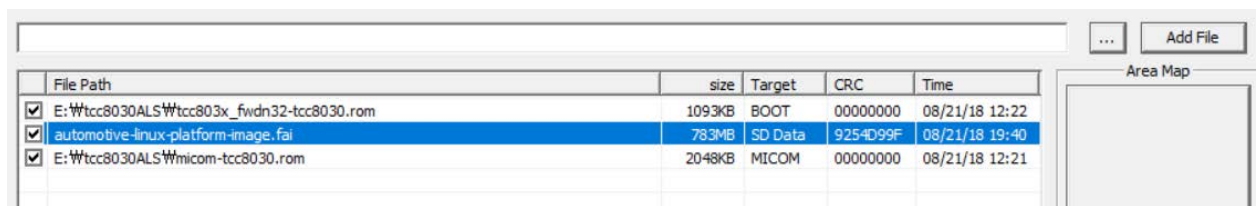


Figure 4.20 Automatically add the FAI File

4.4.6.1.1 SD Data Partition Information

Partition information of SD Data is shown in the table below.

Table 4.4 SD Data Partition Information

FWDN Partition	Size (KB)	Label Name	Class	Block Device eMMC	Description	File Name
Partition 1	102400	secure	Image	mmcblk0p1	Secure Firmware	-
Partition 2	20480	boot	Image	mmcblk0p2	Kernel image	tc-boot-tcc8030.img
Partition 3	1048576	system	Image	mmcblk0p3	Root file system	automotive-linux-platform-image-tcc8030.ext4
Partition 4	1024	env	Image	mmcblk0p4	Environment	-
Partition 5	20480	recovery	Image	mmcblk0p5	Recovery kernel image	tc-boot-tcc8030.img
Partition 6	1024	misc	Image	mmcblk0p6	MISC	-
Partition 7	40960	splash	Image	mmcblk0p7	Parking guide line image	splash.img
Partition 8	512000	home	Image	mmcblk0p8	RW file system for home directory	home-directory.ext4
Partition 9	100	dtb	Image	mmcblk0p9	Device tree	zImage-tcc8030-linux-lpd4321_sv0.1.dtb
Partition 10	Remainder	data	FAT	mmcblk0p10	User storage	-

4.4.6.2 Writing ALS Firmware

To write ALS Firmware to the board, add the u-boot image, fai file, micom_tcc8030.rom image, and select low format whole memory before download, and then click the Start button.

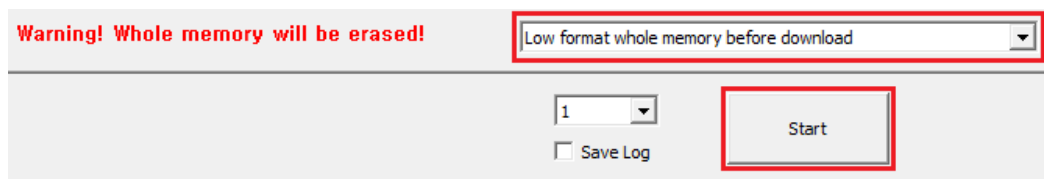


Figure 4.21 Writing ALS Firmware

The following figure shows that ALS Firmware is normally written to the board after clicking the **Start** button.

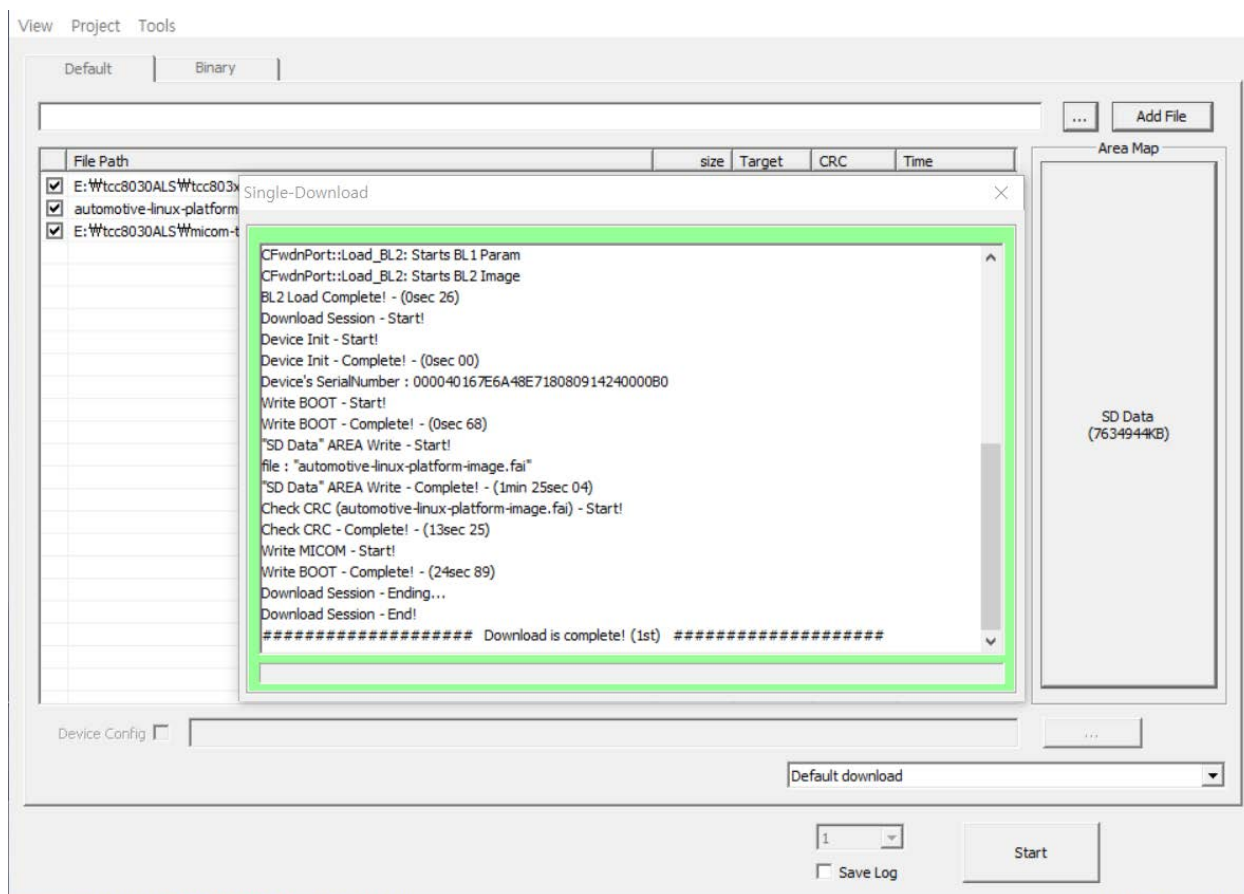


Figure 4.22 Complete write ALS Firmware

4.4.6.3 Firmware Write Option

Firmware Write Option has 2 items as below; Default download and Low format whole memory before download.

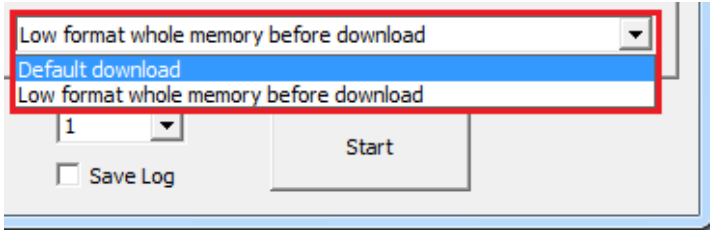


Figure 4.23 Firmware Write Option

4.4.6.3.1 Default Download

Select this option entry to update only certain images while preserving existing partition information. Following shows how to update a specific image.

- ① Bootloader Image Update
After adding new image for 'Bootloader Image Update', click **Start** button to write.

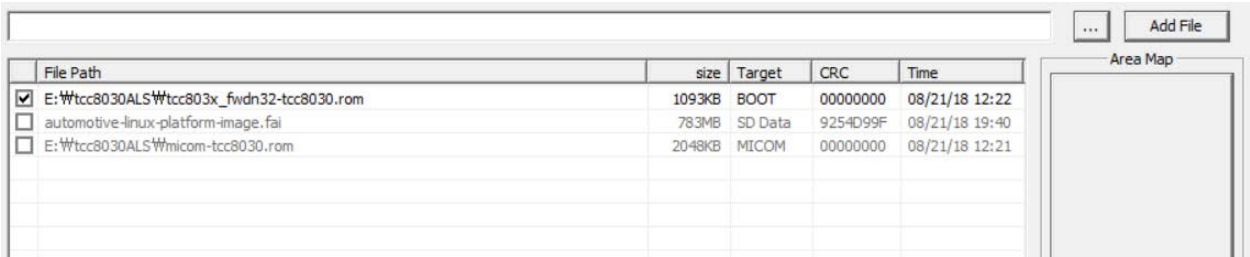


Figure 4.24 Bootloader Image Update

- ② FAI Update

To update one specific image, you must create the FAI file by adding the image to be updated to the partition and then click Start button.
The Figure 4.25 shows that the FAI partition is configured to update the Linux kernel image and the DTB image. As shown in the figure, only certain images to be updated are added to the Image File Path, and the image to be maintained changes to Image File Path blank.

Partition 1	Partition Size	102400 KB (1MB=1024)	Image
	Partition Label	secure	
	Image File Path		
Partition 2	Partition Size	20480 KB (1MB=1024)	Image
	Partition Label	boot	
	Image File Path	tc-boot-tcc8030.img	
Partition 3	Partition Size	1048576 KB (1MB=1024)	Image
	Partition Label	system	
	Image File Path		
Partition 4	Partition Size	1024 KB (1MB=1024)	Image
	Partition Label	env	
	Image File Path		
Partition 5	Partition Size	20480 KB (1MB=1024)	Image
	Partition Label	recovery	
	Image File Path	tc-boot-tcc8030.img	
Partition 6	Partition Size	1024 KB (1MB=1024)	Image
	Partition Label	misc	
	Image File Path		
Partition 7	Partition Size	102400 KB (1MB=1024)	Image
	Partition Label	snapshot	
	Image File Path		
Partition 8	Partition Size	512000 KB (1MB=1024)	Image
	Partition Label	home	
	Image File Path		
Partition 9	Partition Size	100 KB (1MB=1024)	Image
	Partition Label	dtb	
	Image File Path	zImage-tcc8030-linux-lpd4321_sv0.1.dtb	
Partition 10	Partition Size	5887866 KB (1MB=1024)	Image
	Partition Label		
	Image File Path		

Figure 4.25 Configuring Partition for FAI Update

4.4.6.3.2 Low format whole memory before download

It is used to delete the existing information on all partitions and then to write Firmware.

5 PARTITION UPDATE

5.1 Fastboot Flash Memory update

5.1.1 What is Fastboot

Fastboot is a function of ADB (Android Debug Bridge) which is used to update partition content of flash memory in board at bootloader. Functions such as Flash, Erase, and Reboot can be performed according to each partition part of flash memory. The ADB USB Driver must be installed in order to use Fastboot.

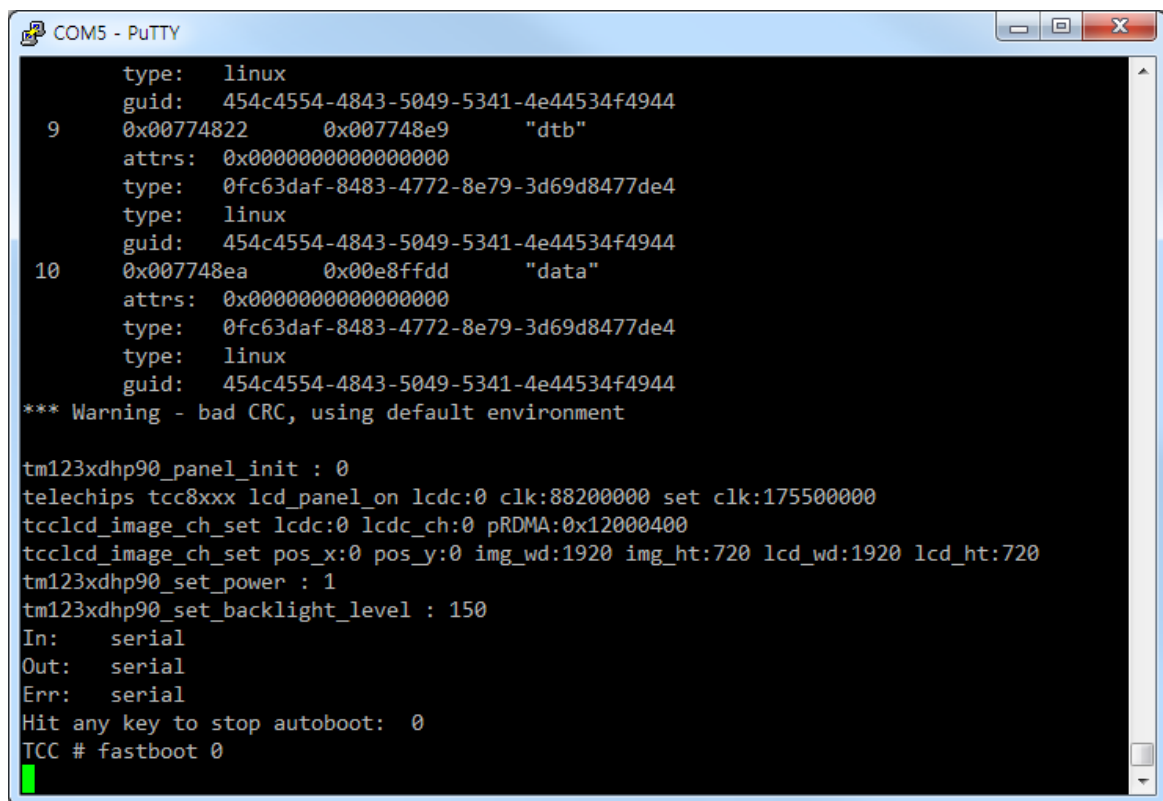
In ALS, all partitions excluding bootloader(u-boot) can be partially updated.

5.1.2 Fastboot Mode

In order to enter Fastboot mode, bootloader must be written using FWDN. If bootloader is already written, follow the procedure below. Use ALS_V3.0.2 or higher version for bootloader.

After bootloader (u-boot) is updated, change boot switch to SNOR boot mode.

- ① Change Boot Switch to SNOR boot mode.
- ② Connect UART serial cable and open UART Console window.
- ③ Supply Power and press 'Enter' key in Console window.
- ④ Run "fastboot 0" in command window.
- ⑤ Fastboot mode will be shown as below.



```

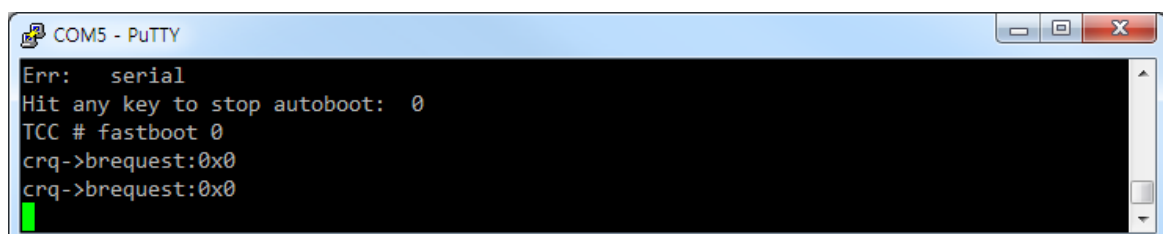
COM5 - PUTTY
type: linux
guid: 454c4554-4843-5049-5341-4e44534f4944
9 0x00774822 0x007748e9 "dtb"
attrs: 0x0000000000000000
type: 0fc63daf-8483-4772-8e79-3d69d8477de4
type: linux
guid: 454c4554-4843-5049-5341-4e44534f4944
10 0x007748ea 0x00e8ffdd "data"
attrs: 0x0000000000000000
type: 0fc63daf-8483-4772-8e79-3d69d8477de4
type: linux
guid: 454c4554-4843-5049-5341-4e44534f4944
*** Warning - bad CRC, using default environment

tm123xdhp90_panel_init : 0
telechips tcc8xxx lcd_panel_on lcdc:0 clk:88200000 set clk:175500000
tcclcd_image_ch_set lcdc:0 lcdc_ch:0 pRDMA:0x12000400
tcclcd_image_ch_set pos_x:0 pos_y:0 img_wd:1920 img_ht:720 lcd_wd:1920 lcd_ht:720
tm123xdhp90_set_power : 1
tm123xdhp90_set_backlight_level : 150
In: serial
Out: serial
Err: serial
Hit any key to stop autoboot: 0
TCC # fastboot 0

```

Figure 5.1 Run fastboot

- ① Connect USB cable to USB2.0_DRD port.
- ② If Android USB Driver is not installed on PC, install Driver.
- ③ Once fastboot is processed, processing commands appear.



```

COM5 - PUTTY
Err: serial
Hit any key to stop autoboot: 0
TCC # fastboot 0
crq->brequest:0x0
crq->brequest:0x0

```

Figure 5.2 Connected fastboot

5.1.3 Installing a USB Driver

- Downloading a USB Driver: <https://developer.android.com/studio/run/win-usb#download>
- Downloading a Android SDK Platform-Tools: <https://developer.android.com/studio/releases/platform-tools>
- Installing a USB Driver: <http://developer.android.com/tools/extras/oem-usb.html#InstallingDriver>

5.1.4 Fastboot flash memory update

Partition information of TCC8030 is as follows.

Table 5.1 TCC8030 Partition

Partition Name	Size	Type	Block Device	Description	File Name
secure	102400	Image	mmcblk0p1	Related secure image	
boot	20480	Image	mmcblk0p2	Kernel image	tc-boot-(MACHINE).img
system	1048576	Image	mmcblk0p3	Root File System	linux-platform-image-(MACHINE).ext4
env	1024	Image	mmcblk0p4	Environment	
recovery	20480	Image	mmcblk0p5	Recovery Kernel image	tc-boot-(MACHINE).img
misc	1024	Image	mmcblk0p6	MISC	
splash	40960	Image	mmcblk0p7	Parking Guide Line Image	splash.img
home	512000	Image	mmcblk0p8	RW file system for Home directory	home-directory.ext4
dtb	100	Image	mmcblk0p9	Device Tree	zImage-(MACHINE)-linux-lpd4321_sv0.1.dtb
Data	remainder	FAT	mmcblk0p10	User Storage	

Use fastboot command in Window PC or Linux PC to write, erase, and reboot Partition.

- fastboot partition write
 - fastboot flash <partition name> <image file name>

Table 5.2 Partition write example

```
fastboot flash boot z:\image\tcc8030\tc-boot-tcc8030.img
fastboot flash recovery z:\image\tcc8030\tc-boot-tcc8030.img
fastboot flash system z:\image\tcc8030\automotive-linux-platform-image-tcc030.ext4
fastboot flash splash z:\image\tcc8030\splash.img
fastboot flash home z:\image\tcc8030\home-directory.ext4
fastboot flash dtb z:\image\tcc030\ Image-tcc8030-linux-lpd4321_sv0.1.dtb
```

- fastboot partition erase
 - fastboot erase <partition name>
- board rebooting
 - fastboot reboot

6 BOOTING AND RUNNING GUIDE

This chapter provides basic explanation regarding 803x EVB booting and ALS application.

6.1 Board Booting Guide

6.1.1 Booting Screen

Once power is supplied to board for normal booting, 2-steps screen conversion is shown as below.

- U-boot screen -> launcher screen

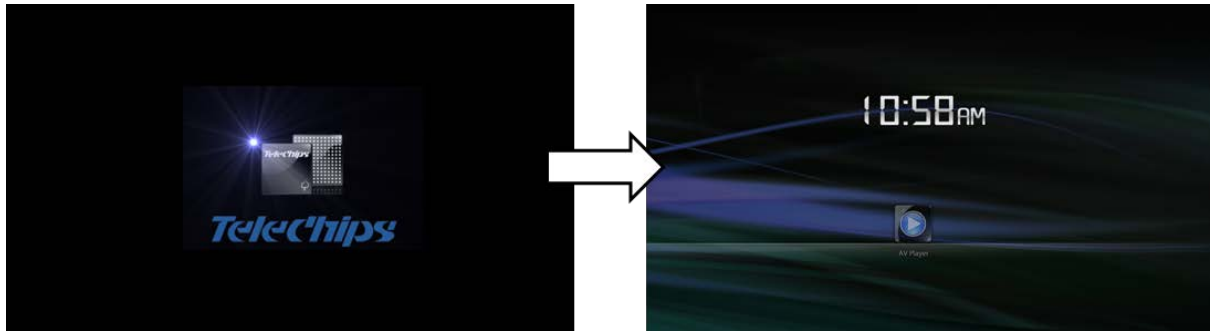


Figure 6.1 Conversion to Booting screen

6.1.2 Linux Console Login

Once Firmware write is completed, a user can log in the Linux Console.

Caution: When resetting, make sure to remove USB cable in USB DRD port in order to protect board and host PC.

Connect UART serial cable to check console screen on PC.

- The UART Configure is as follows;
 - Baudrate: 115200bps
 - Data bits: 8
 - Stop bits: 1
 - Parity: None
- Login ID and Password are "root".

```
[ 0.111926] CPU2: failed to boot: -22
[ 0.131947] CPU3: failed to boot: -22
[ 0.248509] could not find telechips,fbdisplay_port
[ 0.478347] could not find vige_video vige1 node
MAILBOX MSG_EARLYCAMERA_STOP
[ 0.594574] tcc_cpufreq_probe: failed to get regulator_get for vdd_cp
u0
Mounting procfs
Mounting sysfs
sh: 1: unknown operand
System time before build time, advancing clock.
Failed to insert module 'autofs4': No such file or directory

Telechips INVITE Baseline (Poky/meta-telechips/meta-core) 5.0.0 telechip
s-tcc8021 ttyAMA0

telechips-tcc8021 login:
Telechips INVITE Baseline (Poky/meta-telechips/meta-core) 5.0.0 telechip
s-tcc8021 ttyAMA0

telechips-tcc8021 login: root
Password:
root@telechips-tcc8021:~#
root@telechips-tcc8021:~#
```

Figure 6.2 Console Log in

6.2 System Architecture

6.2.1 Block Diagram

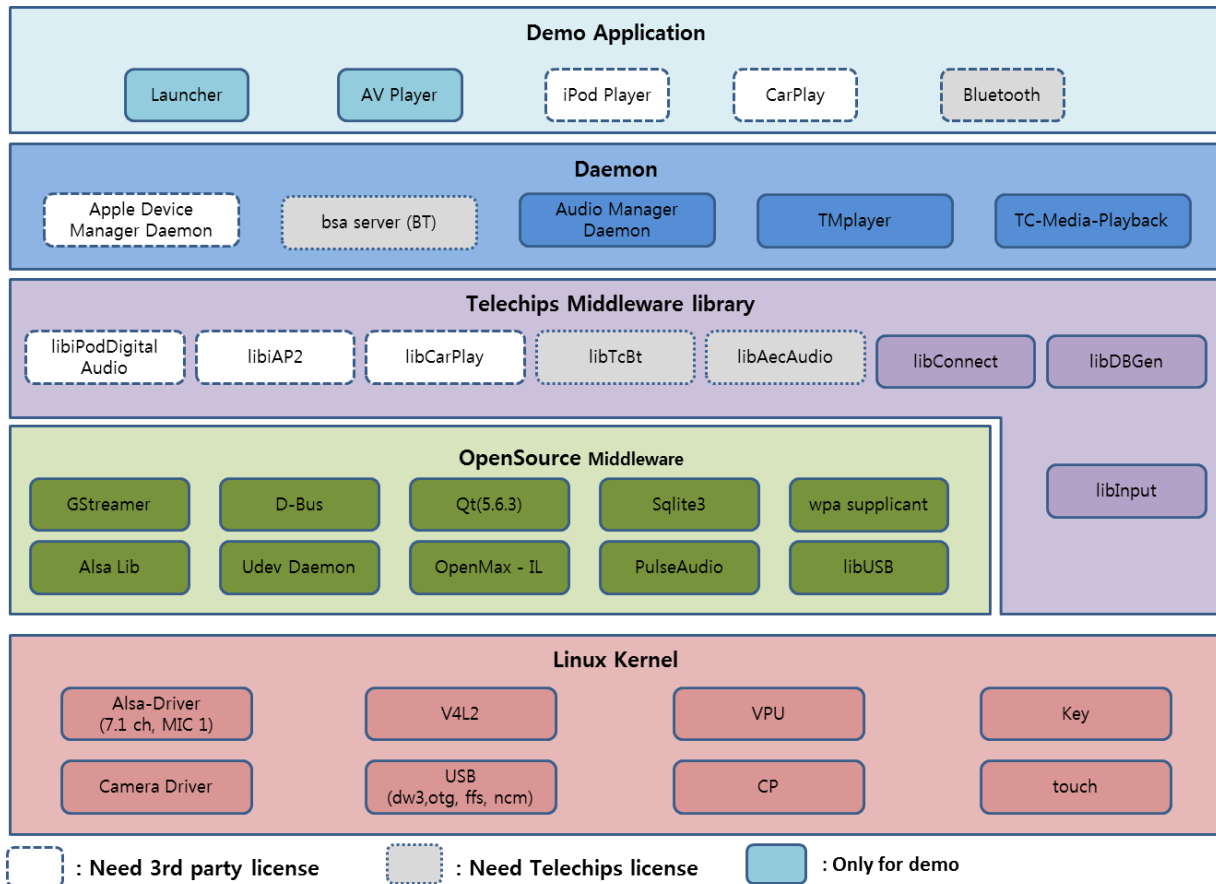


Figure 6.3 ALS Architecture

6.2.2 Structure

TCC Automotive Linux SDK consists of Linux kernel, Middleware, Daemon, and Demo Application.

6.2.2.1 Linux Kernel Layer

Linux kernel Layer includes Telechips' BSP for Automotive.

6.2.2.2 Middleware

Middleware is composed of Open Source Middleware (OSM) and Telechips Middleware Library. For OSS package list required for Automotive Linux SDK, refer to the exhibits.

Telechips Middleware Libraries are as follows.

- libConnect: Library to check Insert/Remove, mount/unmount of a removable device
- libDBGGen: Library to make File / Meta DB
- libiPodDigitalAudio, libiAP2: Library for iPod play
- libCarPlay: Library for CarPlay
- libTcBt, libAecAudio: Library for BT
- libInput: Library for an input device such as key or touch

6.2.2.3 Daemon

- Audio Manager Daemon: It is a daemon to control Audio in In-Vehicle Infotainment.
- Apple Device Manager Daemon: It is a daemon to operate connection/disconnection on apple devices.
- bsa server(BT): It is a daemon to operate BT.
- tc-media-playback: It is a daemon to A/V play (using Gstreamer).
- tmplayer: It is a daemon to control tc-media-playback and media database.

6.2.2.4 Application

The followings are Demo Applications provided by Telechips.

- AV Player
- iPod Player
- CarPlay
- Bluetooth

6.2.3 Motion Scenario

Once booting is completed, menu icon will appear on launcher screen. For execution, follow the procedures below.

- AV Player
 - Connecting Storage device to port enables auto play.
 - USB device: Insert to USB Host or DRD port
 - SDMMC: Insert to SD slot
- iPod Connection
 - Insert iPod device to USB Host or DRD port.
- CarPlay
 - Insert device for CarPlay to USB DRD port.
 - Only USB DRD port is available for CarPlay.

Refer to following chapters for detailed methods to use each application.

For details about CarPlay or iAP, refer to the release note of CarPlay/iAP2 SDK which is released separately.

6.3 ALS Application Guide

This chapter is for development of In-vehicle multimedia player by using ALS

6.3.1 Available Documentation

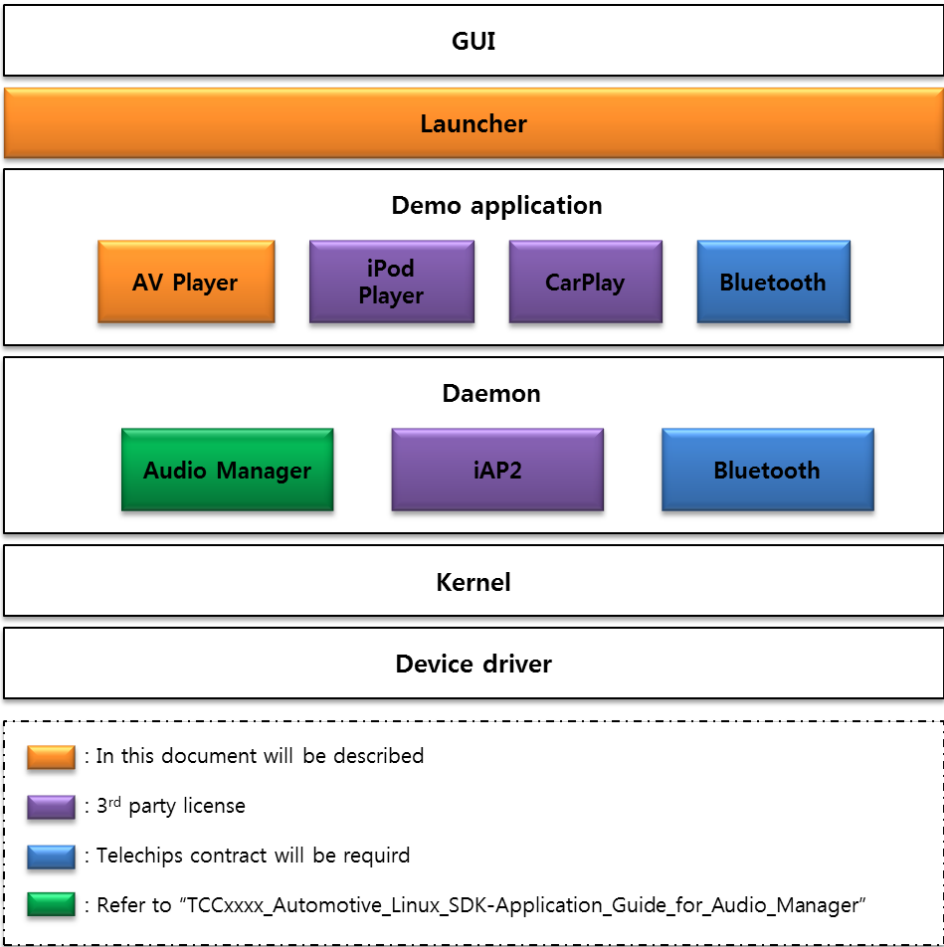


Figure 6.4 Document range

6.3.2 Launcher

6.3.2.1 Overview

Launcher executes various applications and executes simple display management via DBus message.

6.3.2.2 Block Diagram

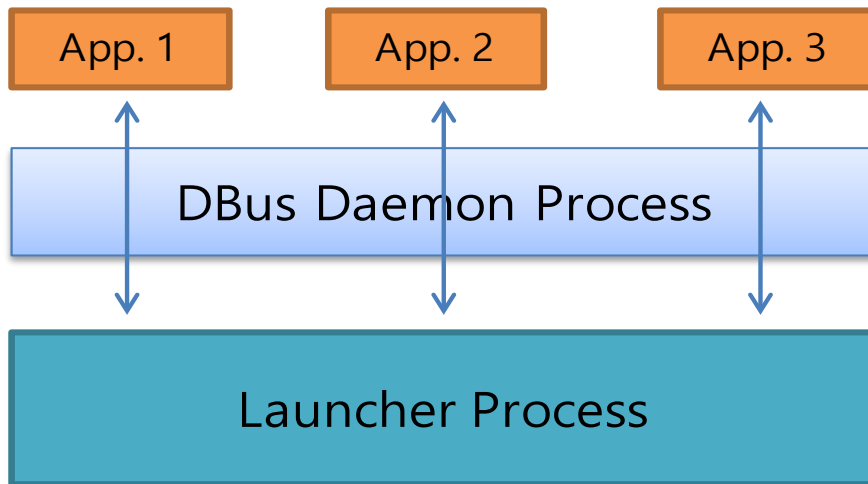


Figure 6.5 Launcher block diagram

6.3.2.3 Feature

- Execute Application: It executes applications defined in the environment setting files.
- Show/Hide Application: It executes application show/hide function by using DBus message.

6.3.2.4 Required packages

- Qt 5.6.3
- DBus

6.3.2.5 UI Guide

6.3.2.5.1 Display

When Launcher executes, it executes all registered applications, and only the icons of available application are activated.

6.3.2.5.2 Execute

If you click an activated icon, the concerned application appears in the screen and the launcher disappears from the screen.



Figure 6.6 Launcher GUI

6.3.3 AV Player

This chapter explains the architecture of AV player.

6.3.3.1 Overview

The AV player is a multimedia player to play Audio and Video File in an automotive environment.

The AV player makes database for meta information of media files by recognizing a storage device and includes playing media files and GUI.

6.3.3.1.1 Block Diagram

The block diagram of AV player is as follows.

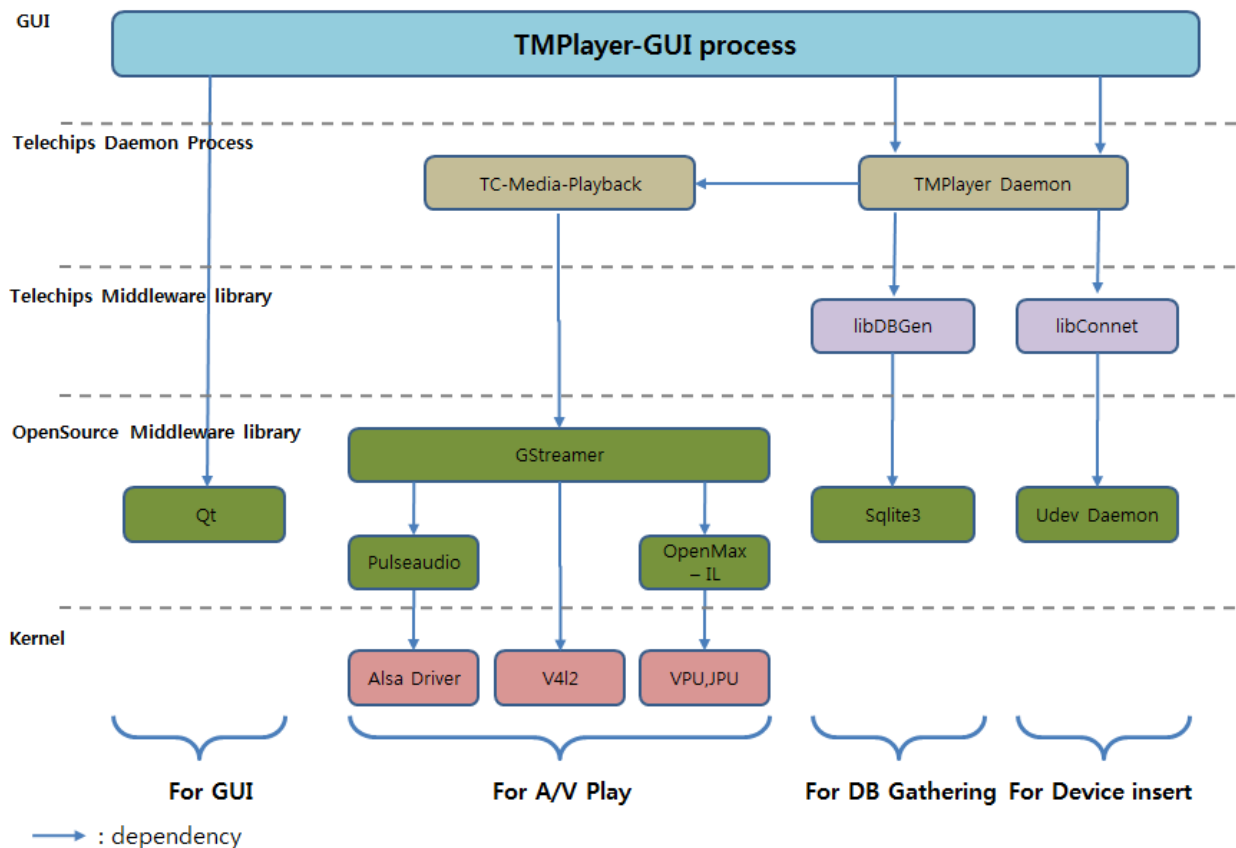


Figure 6.7 AV player Block Diagram

AV player is based on open source such as Qt, GStreamer, Sqlite3, and Udev.

Telechips Middleware library includes necessary functions in the multimedia player by using each open source APIs.

Each Telechips Middleware library is as follows.

- libConnect: Check the status change of a removable device (i.e. USB, SDMMC, and etc) through Udev.
- libDBGen: Search available multimedia contents and proceed database for the contents by using Sqlite3.

AV player process includes GUI and a scenario for demonstration.

6.3.3.1.2 Flow

The AV player plays Audio and Video content (only file base) with the following procedure.

- ① Status check of a storage device such as USB or SDMMC.
- ② Build database with Mount path.
- ③ Generate a play list based on the database.
- ④ Play an audio/video file through Gstreamer.

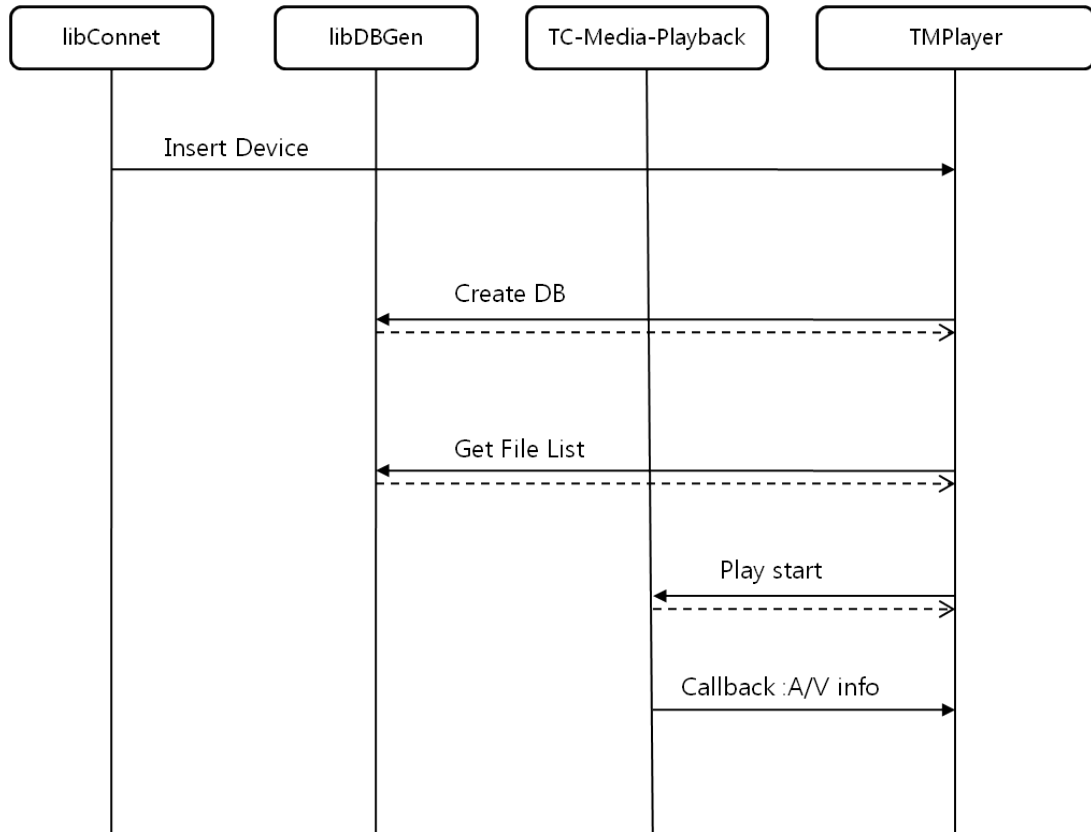


Figure 6.8 AV player Sequence diagram

6.3.3.2 Functions

6.3.3.2.1 Feature

- Audio content play
- Video content play
- Supports File/Meta DB Browsing
- Supports file seek and pause/resume
- Supports shuffle/repeat mode
- Supports device: USB, and SDMMC
- Supports multi device and device change
 - Three USBs and one SDMMC can be simultaneously mounted and a device can be selected by mode change.
- Auto play when a New Device is inserted
 - If only audio file exists, audio auto play is proceeded.
 - If only video file exists, video auto play is proceeded.
- For support codec list, refer to "TCCxxx Automotive Linux SDK-Reference Manual for Supported Media Format".

6.3.3.2.2 Required packages

- Qt 5.6.3
- GStreamer 1.10.4
- V4l2 for video output (V4l2sink of Gstreamer)
- hardware vpu
- alsa-lib

6.3.3.3 GUI Guide

GUI of AV play consists of audio and video screen.

If there is only audio file in a storage device, it is not available to switch to video UI. In case that there is only a video file, it is not available to switch to audio UI.

6.3.3.3.1 Audio Play

The following image shows audio GUI.

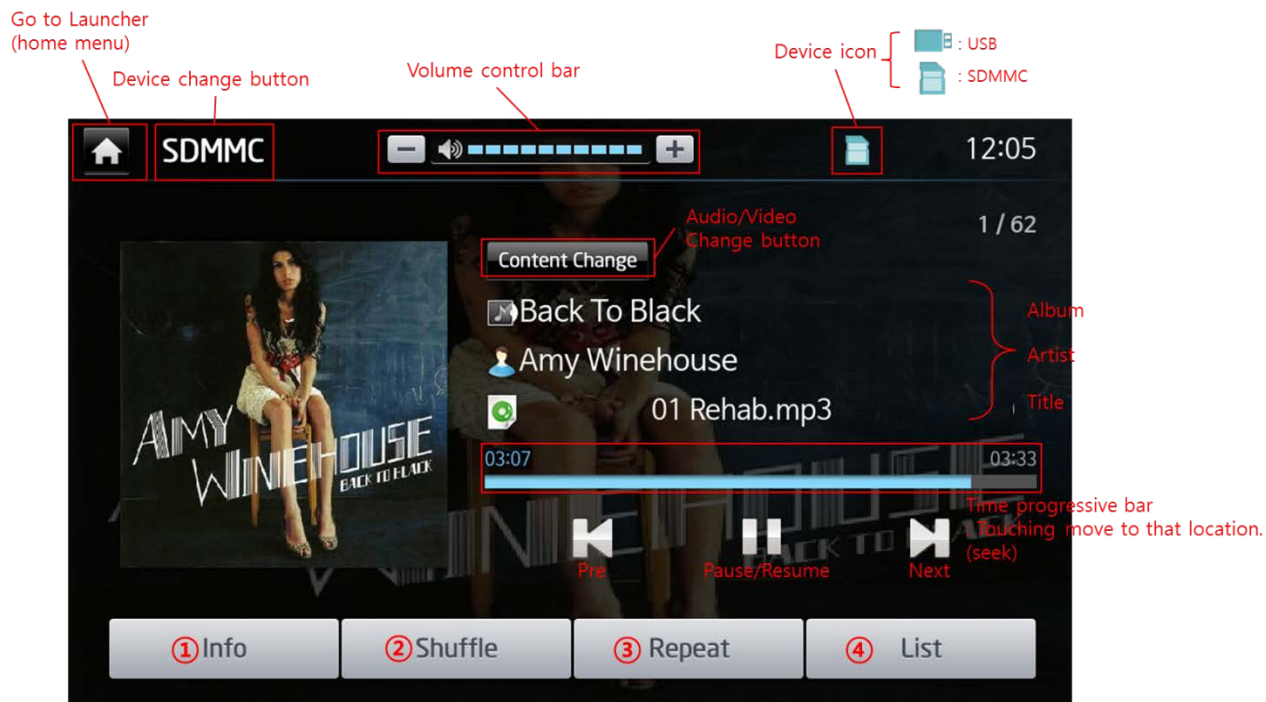






Figure 6.9 Audio Play Screen

■ Info

- If Button ① is selected, meta information of a currently played file appears in the pop-up window.



Figure 6.10 Audio Information Pop-up

- Shuffle
 - Button ② shuffles the play list of the current file.
 - Non-shuffle (normal mode), Folder shuffle, and All shuffle are supported and an icon is displayed at the top of the screen per each mode.
 - Folder shuffle
 
 - All shuffle
 
- Repeat
 - Button ③ repeats the play list of the current file. Each mode is as follows.
 - One song repeat
 
 - Folder repeat
 
- List
 - Button ④ shows the play list.
 - The play list provides the file list and meta list.
 - The file list is the play list sorted with a file name and it can change depending on shuffle or repeat.

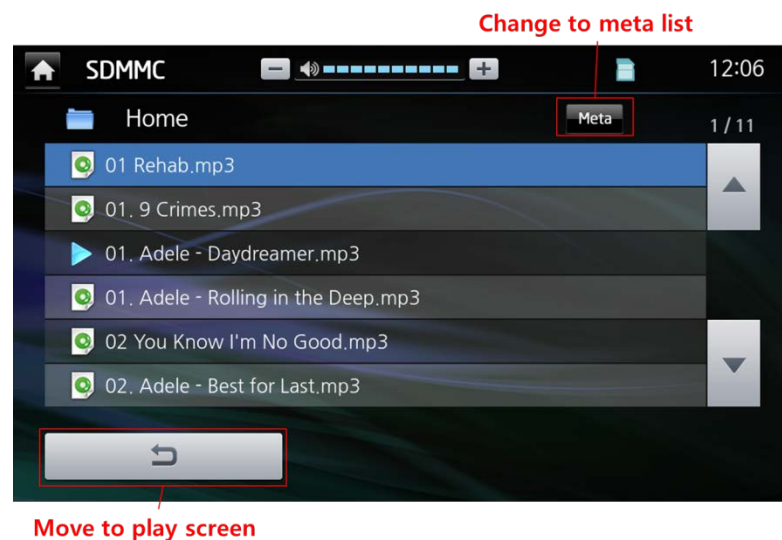


Figure 6.11 Audio File List Screen

Meta list is sorted per meta category. It is a tree structure and a sub-category can be selected. Meta list is composed of top 4 categories as Figure 6.12.

Each category has the following sub-category.

- Genre -> Artist -> Album -> Track
- Artist -> Album -> Track
- Album -> Track
- Track

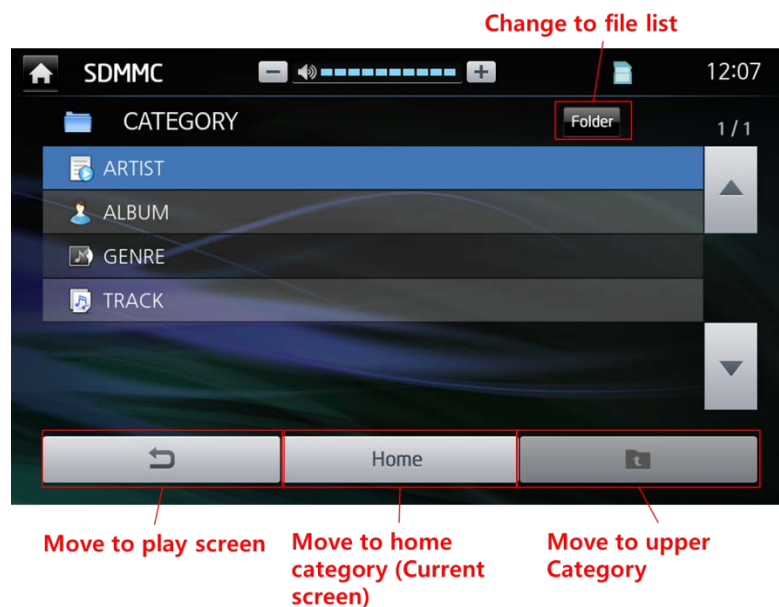


Figure 6.12 Audio Meta List Screen

6.3.3.3.2 Video Play

Video Play GUI is as follows.

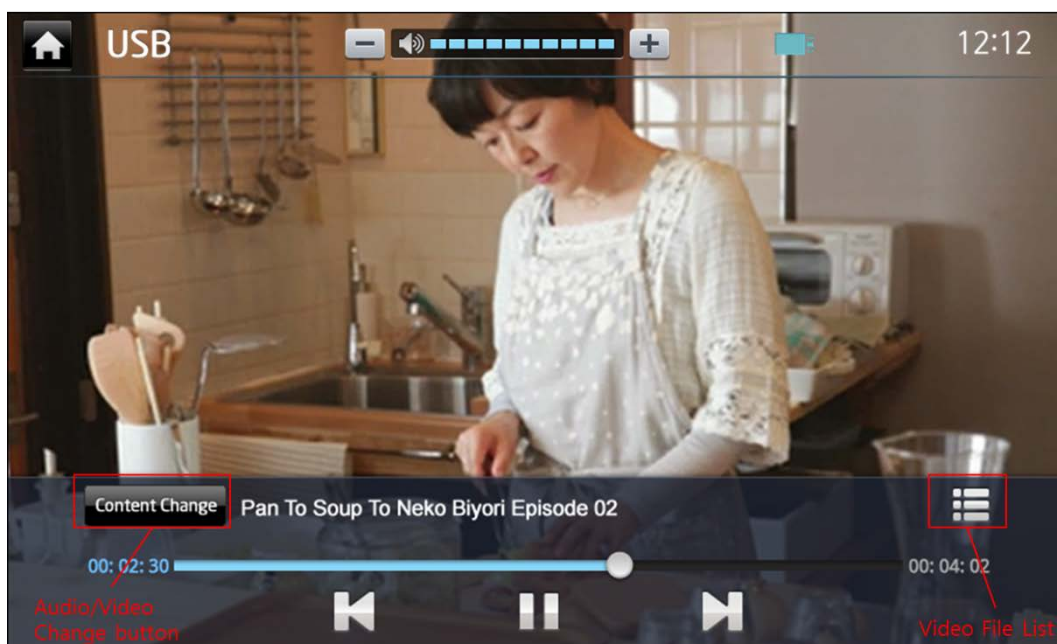


Figure 6.13 Video Play Screen

For details of Yocto Project, refer to following documents;

7 REFERENCE

- [1] Yocto Project Quick Start; <https://www.yoctoproject.org/docs/2.4.2/yocto-project-qs/yocto-project-qs.html>
- [2] Yocto Project Wiki; https://wiki.yoctoproject.org/wiki/Main_Page
- [3] Yocto Project Development Manual; <https://www.yoctoproject.org/docs/2.4.2/dev-manual/dev-manual.html>
- [4] Yocto Project Reference Manual; <https://www.yoctoproject.org/docs/2.4.2/ref-manual/ref-manual.html>
- [5] Yocto Project Application Developer's Guide; <https://www.yoctoproject.org/docs/2.4.2/adt-manual/adt-manual.html>
- [6] BitBake User Manual; <https://www.yoctoproject.org/docs/2.4.2/bitbake-user-manual/bitbake-user-manual.html>

8 REVISION HISTORY

8.1 Rev. 2.10: 2018-09-05

- Table 5.1 and Table 5.2 are updated
- Snapshot partition is removed
- Build Guide, FWDN Guide, and Quick Start Guide are merged for usability.

8.2 Rev. 2.01: 2018-06-29

- Typos are corrected.

8.3 Rev. 2.00: 2018-05-31

- TCC8030 Board is added
- Not supported machines are removed

8.4 Rev. 1.08: 2017-09-19

- Application Guide is merged
- Content of firmware download into FWDN guide document are separated

8.5 Rev. 1.07: 2016-09-07

- 8021 FWDN Boot mode is changed

8.6 Rev. 1.06: 2016-08-10

- Partition Mode – GPT is added
- TCC8925-LCN, and TCC8021-EVB are added
- TCC8925-EVB is removed

8.7 Rev. 1.05: 2016-04-26

- Snapshot partition size and Kernel image name are changed
- Fastboot flash write guide is added

8.8 Rev. 1.04: 2015-11-30

- Figures (Bootling Screen Switch, TCC8935 LCN USB Cable Connection Screen) are changed

8.9 Rev. 1.03: 2015-09-16

- FWDN guide is updated

8.10 Rev. 1.02: 2015-08-31

- Supported boards as TCC8925/30/31/60/71 are added.

8.11 Rev. 1.01: 2015-05-27

- Typos are corrected

8.12 Rev. 1.00: 2015-03-31

- First version release

DISCLAIMER

All information and data contained in this material are without any commitment, are not to be considered as an offer for conclusion of a contract, nor shall they be construed as to create any liability. Any new issue of this material invalidates previous issues. Product availability and delivery are exclusively subject to our respective order confirmation form; the same applies to orders based on development samples delivered. By this publication, Telechips, Inc. does not assume responsibility for patent infringements or other rights of third parties that may result from its use.

Further, Telechips, Inc. reserves the right to revise this publication and to make changes to its content, at any time, without obligation to notify any person or entity of such revisions or changes.

No part of this publication may be reproduced, photocopied, stored on a retrieval system, or transmitted without the express written consent of Telechips, Inc.

This product is designed for general purpose, and accordingly customer be responsible for all or any of intellectual property licenses required for actual application. Telechips, Inc. does not provide any indemnification for any intellectual properties owned by third party.

Telechips, Inc. can not ensure that this application is the proper and sufficient one for any other purposes but the one explicitly expressed herein. Telechips, Inc. is not responsible for any special, indirect, incidental or consequential damage or loss whatsoever resulting from the use of this application for other purposes.

COPYRIGHT STATEMENT

Copyright in the material provided by Telechips, Inc. is owned by Telechips unless otherwise noted.

For reproduction or use of Telechips' copyright material, permission should be sought from Telechips. That permission, if given, will be subject to conditions that Telechips' name should be included and interest in the material should be acknowledged when the material is reproduced or quoted, either in whole or in part. You must not copy, adapt, publish, distribute or commercialize any contents contained in the material in any manner without the written permission of Telechips. Trade marks used in Telechips' copyright material are the property of Telechips.

Important Notice

This material may include technology owned by the 3rd party licensor and the technology may be subject to its associated licenses. It is solely customer's responsibility to identify and comply with such licenses. No other licenses are granted or implied by Telechips with making available this material.

For customers who use licensed Codec ICs and/or licensed codec firmware of mp3:

"Supply of this product does not convey a license nor imply any right to distribute content created with this product in revenue-generating broadcast systems (terrestrial. Satellite, cable and/or other distribution channels), streaming applications(via internet, intranets and/or other networks), other content distribution systems(pay-audio or audio-on-demand applications and the like) or on physical media(compact discs, digital versatile discs, semiconductor chips, hard drives, memory cards and the like). An independent license for such use is required. For details, please visit <http://mp3licensing.com>".

For customers who use other firmware of mp3:

"Supply of this product does not convey a license under the relevant intellectual property of Thomson and/or Fraunhofer Gesellschaft nor imply any right to use this product in any finished end user or ready-to-use final product. An independent license for such use is required. For details, please visit <http://mp3licensing.com>".

For customers who use Digital Wave DRA solution:

"Supply of this implementation of DRA technology does not convey a license nor imply any right to this implementation in any finished end-user or ready-to-use terminal product. An independent license for such use is required."

For customers who use DTS technology:

"This product made under license to certain U.S. patents and/or foreign counterparts."
"© 1996 – 2011 DTS, Inc. All rights reserved."

For customers who use Dolby technology:

"Supply of this Implementation of Dolby technology does not convey a license nor imply a right under any patent, or any other industrial or intellectual property right of Dolby Laboratories, to use this Implementation in any finished end-user or ready-to-use final product. It is hereby notified that a license for such use is required from Dolby Laboratories."

For customers who use Google technology:

"Copyright © 2013 Google Inc. All rights reserved."

For customers who use MS technology:

"This product is subject to certain intellectual property rights of Microsoft and cannot be used or distributed further without the appropriate license(s) from Microsoft."