Tinalinux

SDK 开发指南

1.0

2019.02.27

文档履历

版本号	日期	制/修订人	内容描述
0.1	2019.02.20	AW1225	创建
1.0	2019.02.27	AW1225	正式发布

目录

1. 概述

1.1 编写目的

本文档作为 Allwinner Tina Linux 系统平台开发指南，旨在帮助软件开发工程师、技术支持工程师快速上手，熟悉 Tina Linux 系统的开发及调试流程。

1.2 适用范围

Tina Linux v3.5 及以上版本。

1.3 相关人员

本开发指南适用于 Tina 系统软件开发工程师、Tina 系统技术支持工程师。

2. Tina 系统资料

2.1 概述

Tina SDK 发布的文档旨在帮助开发者快速上手开发及调试，文档中涉及的内容并不能涵盖所有的开发知识和问题。文档列表也正在不断更新，如有文档上的疑问及需求，请联系 Allwinner FAE 窗口、或发送邮件到 tina-linux@allwinnertech.com。

Tina SDK 提供丰富的文档资料，文档保存在 ``TinaSDK/docs/" 目录下，包括硬件参考设计文档、Flash 支持列表、量产工具使用说明、软件开发与制定介绍文档、芯片研发手册等资料。

2.2 文档列表

2.2.1 硬件文档

硬件文档路径在 ``TinaSDK/docs/硬件文档" 目录，包含以下文档。

硬件文档	1	标案原理图	
	2	标案硬件设计说明书	包括原理图、PCB和散热的说明
	3	PCB参考设计	

图 1: TinaLinux 硬件文档

2.2.2 支持列表

支持列表文档在 ``TinaSDK/docs/支持列表" 目录。包括以下文档。

支持列表	1	SPI Nand支持列表	
	2	Nor Flash支持列表	
	3	WiFi_BT支持列表	

图 2: TinaLinux 支持列表

2.2.3 工具文档

工具文档在 ``TinaSDK/docs/工具文档" 目录，包括以下文档。

工具文档	1	PhoenixSuit使用说明书	APST上工具使用说明
	2	PhoenixPro使用说明书	
	3	DragonSN使用说明书	
	4	DragonMAT使用说明书	
	5	DragonFace-cdr说明书	
	6	DragonHD说明书	

图 3: TinaLinux 工具文档

2.2.4 IC 文档

IC 文档在 ``TinaSDK/docs/IC 文档" 目录，包括以下文档。

IC文档	1	IC Brief	
	2	IC Datasheet	
	3	IC User Manual	

图 4: TinaLinux IC 文档

2.2.5 系统开发指南

软件文档在 ``TinaSDK/docs/系统开发文档" 目录，包括以下文档。

软件文档	1	TinaLinux支持列表_CODEC_Vxx	介绍外挂CODEC支持列表
	2	TinaLinux SDK开发指南	介绍SDK适用平台列表； 提供文档索引与介绍； 介绍开发环境搭建； 介绍SDK安装准备以及编译烧写； 介绍SDK软件结构； 介绍UBOOT/KERNEL/APP定制开发； 介绍系统调试工具； 常见问题说明；
	3	TinaLinux SDK V3.5更新说明文档	介绍版本更新内容
	4	TinaLinux 系统配置说明文档	介绍系统软件配置信息(Tina/kernel config)；系统硬件资源配置(sysconfig/device tree)；介绍分区表配置及分区功能，增加一个章节说明Nor/Nand切换使用配置
	5	TinaLinux OTA开发指南	介绍OTA功能，指导如何配置、启用、定制以及常见问题总结
	6	TinaLinux 功耗管理开发指南	介绍功耗管理功能，指导如何配置、启用、定制以及常见问题总结
	7	TinaLinux 系统裁剪开发指南	介绍如何进行系统裁剪，指导客户使用、定制
	8	TinaLinux 启动优化开发指南	介绍如何进行系统启动优化，指导客户使用、定制
	9	TinaLinux WIFI开发指南	介绍wifi服务、API使用方式以及模组移植指导
	10	TinaLinux Bluetooth开发指南	介绍BT服务、API使用方式以及模组移植指导
	11	TinaLinux Smartlink开发指南	介绍smarklink服务，API使用方式以及移植指导
	12	TinaLinux Audio开发指南	介绍Tina Audio框架、使用接口，通路等，指导用户配置、使用、测试等
	13	TinaLinux Security开发指南	介绍Tina安全功能，指导security boot、security os、selinux以及网络安全等功能使用与开发
	14	TinaLinux tplayer开发和使用指南	介绍多媒体解码功能、上层使用接口，指导如何配置、使用、测试等
	15	TinaLinux trecorder开发和使用指南	介绍多媒体编码、上层使用接口，指导如何配置、使用、测试等
	16	TinaLinux 存储管理开发指南	介绍系统挂载、分区管理、存储设备插拔监测与挂载、存储管理机制等内容
	17	TinaLinux 系统调试指南	介绍Tina linux常用调试工具，指导用户配置、使用
	18	WIFI RF工具使用说明文档	Tina WIFI RF测试、使用指导
	19	Tina 量产测试使用文档	Tina 量产测试配置，使用指导
	20	平台功耗说明文档	Tina 各硬件平台功耗数据
	21	Flash性能说明文档	Tina FLSAH性能报告
	22	WIFI/BT性能说明文档	Tina WIFI/BT模组性能报告
	23	TinaLinux SDK压力测试方法	Tina SDK压力测试报告(CPU/DRAM带宽)
	24	语音AI engin使用说明文档	语音AI engin使用说明文档
	25	Tina LED灯效配置说明文档	Tina LED灯效配置、使用指导

图 5: TinaLinux 系统开发文档

2.2.6 BSP 文档列表

Linux BSP 开发文档在 ``TinaSDK/docs/BSP 开发" 目录，包括以下文档。

BSP文档	24	Audio	开发说明（配置、使用）、调试文档
	25	CLK	开发说明（配置、使用）、调试文档
	26	DDR	开发说明（配置、使用）、调试文档
	27	DEBUG	开发说明（配置、使用）、调试文档
	28	DVFS	开发说明（配置、使用）、调试文档
	29	DMA	开发说明（配置、使用）、调试文档
	30	I2C	开发说明（配置、使用）、调试文档
	31	PWM	开发说明（配置、使用）、调试文档
	32	PINCTRL	开发说明（配置、使用）、调试文档
	33	SECURITY	开发说明（配置、使用）、调试文档
	34	SPI	开发说明（配置、使用）、调试文档
	35	THERMAL	开发说明（配置、使用）、调试文档
	36	UBOOT	开发说明（配置、使用）、调试文档
	37	UART	开发说明（配置、使用）、调试文档
	38	USB	开发说明（配置、使用）、调试文档
	39	watchdog	开发说明（配置、使用）、调试文档
	40	interrupt	开发说明（配置、使用）、调试文档

图 6: TinaLinux BSP 开发文档

3. Tina 系统概述

3.1 概述

Tina Linux 系统是基于 openwrt-14.07 的版本的软件开发包，包含了 Linux 系统开发用到的内核源码，驱动，工具、系统中间件与应用程序包。openwrt 是一个开源的嵌入式 Linux 系统自动构建框架，是由 Makefile 脚本和 Kconfig 配置文件构成的。使得用户可以通过 menuconfig 配置，编译出一个完整的可以直接烧写到机器上运行的 Linux 系统软件。

3.2 系统框图

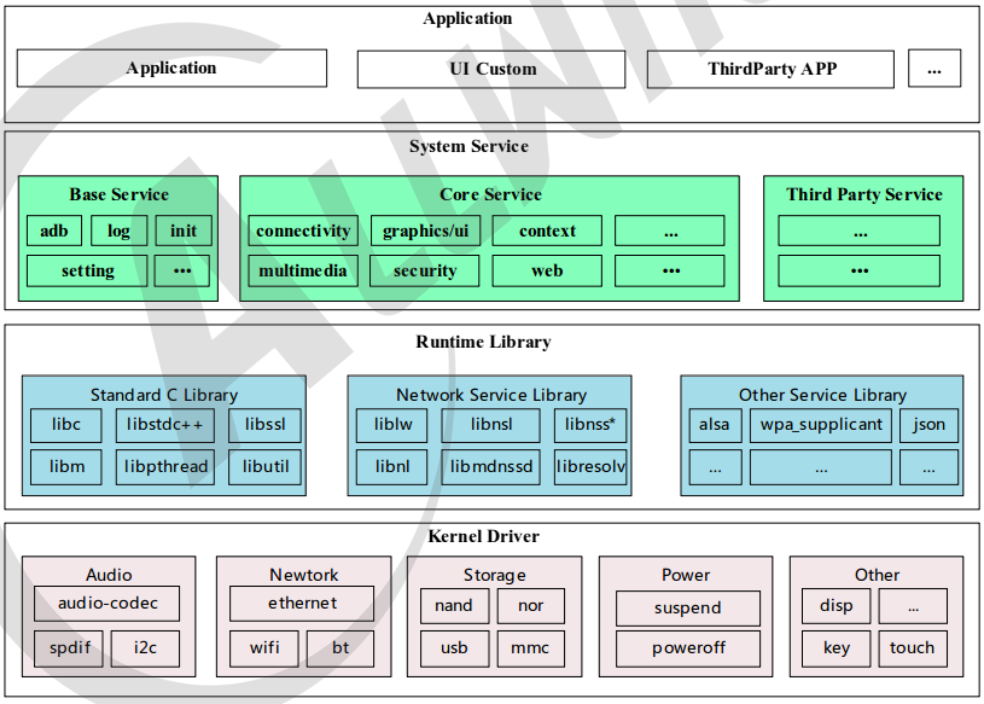

图 7: Tina Linux 系统框图

Tina 系统软件框图如图所示，从下至上分为 Kernel && Driver、Libraries、System Services、Applications 四个层次。各层次内容如下：

1. Kernel&&Driver 主要提供 Linux Kernel 的标准实现。Tina 平台的 Linux Kernel 采用 Linux3.4、linux3.10、linux4.4、linux4.9 等内核 (不同硬件平台可能使用不同内核版本)。提供安全性，内存管理，进程管理，网络协议栈等基础支持；主要是通过 Linux 内核管理设备硬件资源，如 CPU 调度、缓存、内存、I/O 等。

2. Libraries 层对应一般嵌入式系统，相当于中间件层次。包含了各种系统基础库，及第三方开源程序库支持，对应用层提供 API 接口，系统定制者和应用开发者可以基于 Libraries 层的 API 开发新的应用。

3. System Services 层对应系统服务层，包含系统启动管理、配置管理、热插拔管理、存储管理、多媒体中间件等。

4. Applications 层主要是实现具体的产品功能及交互逻辑，需要一些系统基础库及第三方程序库支持，开发者可以开发实现自己的应用程序，提供系统各种能力给到最终用户。

3.3 开发流程

Tina Linux 系统是基于 Linux Kernel，针对多种不同产品形态开发的 SDK。可以基于本 SDK，有效地实现系统定制和应用移植开发。

图 8: Tina Linux 系统开发流程

如上图所示，开发者可以遵循上述开发流程，在本地快速构建 Tina Linux 系统的开发环境和编译代码。下面将简单介绍下该流程：

1. 检查系统需求：在下载代码和编译前，需确保本地的开发设备能够满足需求，包括机器的硬件能力，软件系统，工具链等。目前 Tina Linux 系统只支持 Ubuntu 操作系统环境下编译，并仅提

供 Linux 环境下的工具链支持，其他如 MacOS，Windows 等系统暂不支持。

2. 搭建编译环境：开发机器需要安装的各种软件包和工具，详见 < 第 4.1 章节代码环境搭建 >，获知 TinaLinux 已经验证过的操作系统版本，编译时依赖的库文件等。

3. 选择设备：在编译源码前，开发者需要先导出预定义环境变量，然后根据开发者根据的需求，选择对应的硬件板型，详见 < 第 6.1 章节 Tina 完整系统编译 >。

4. 系统定制：开发者可以根据使用的硬件板子、产品定义，定制 U-Boot(详见 8 章 U-Boot 开)、Kernel(详见 9 章 Kernel 开发) 及 Openwrt(详见 10 章系统开发)，请参考章节中相关开发指南和配置的描述。

5. 编译与打包：完成设备选择、系统定制之后执行编译命令，包括整体或模块编译以及编译清理等工作，进一步的，将生成的 boot/内核二进制文件、根文件系统、按照一定格式打包成固件。(< 第 6.1 章节 Tina 完整系统编译 >)

6. 烧录并运行：继生成镜像文件后，将介绍如何烧录镜像并运行在硬件设备，进一步内容详见 (第 7 章系统烧写)。

4. Tina 开发环境

4.1 概述

嵌入式产品开发流程中，通常有两个关键的步骤，编译源码与烧写固件。源码编译需要先准备好编译环境，而固件烧写则需要厂家提供专用烧写工具。本章主要讲述这如何搭建环境来实现 Tina sdk 的编译、烧写。

4.2 编译环境搭建

一个典型的嵌入式开发环境包括本地开发主机和目标硬件板，

- 本地开发主机作为编译服务器，需要提供 linux 操作环境，建立交叉编译环境，为软件开发提供代码更新下载，代码交叉编译服务。
- 本地开发主机通过串口或 USB 与目标硬件板连接，可将编译后的镜像文件烧写到目标硬件板，并调试系统或应用程序。

4.2.1 开发主机配置

Tina Linux SDK 是在 ubuntu14.04 开发测试的，因此我们推荐使用 **Ubuntu 14.04**主机环境进行源码编译，其他版本没有具体测试，可能需要对软件包做相应调整。

4.2.2 软件包配置

编译 Tina Linux SDK 之前，需要先确定编译服务器安装了 gcc, binutils, bzip2, flex, python, perl, make,ia32-libs, find, grep, diff, unzip, gawk, getopt, subversion, libz-dev, libc headers.

ubuntu 可直接执行以下命令安装：

```
sudo apt-get install build-essential subversion git-core libncurses5-dev zlib1g-dev gawk flex quilt libssl-dev xsltproc libxml-parser-perl mercurial bzr ecj
    cvs unzip ia32-libs -y
```

4.3 固件烧写搭建

TinaLinux SDK 包括固件烧写工具，具体路径在 ``TinaSDK/tools/aw_tools/" 目录下。

- 主机是 windows
  需要使用 PhoenixSuit.zip，软件配置与使用详见工具文档《PhoenixSuit 使用说明书》。
- 主机是 Ubuntu
  64bit 主机使用 LiveSuitV306_For_Linux64.zip，32bit 主机使用 LiveSuitV306_For_Linux32.zip，软件配置与使用详见工具文档《LiveSuit 使用说明书》文档。

5. Tina 系统获取

5.1 概述

5.2 SDK 获取

Allwinner Tina Linux SDK 通过全志代码服务器对外发布。客户需要向业务/技术支持窗口申请 SDK 下载权限。申请需同步提供 SSH 公钥进行服务器认证授权，获得授权后即可同步代码。

5.3 SDK 结构

Tina Linux SDK 主要由构建系统、配置工具、工具链、host 工具包、目标设备应用程序、文档、脚本、linux 内核、bootloader 部分组成，下文按照目录顺序介绍相关的组成组件.

```
Tina-SDK/
    ├── build/
    ├── config/
    ├── Config.in
    ├── dl/
    ├── docs/
    ├── lichee/
    ├── Makefile
    ├── out/
    ├── package/
    ├── prebuilt/
    ├── rules.mk
    ├── scripts/
    ├── target/
    ├── toolchain/
    └── tools/
```

5.3.1 build 目录

build 目录存放 Tina Linux 的构建系统文件，此目录结构下主要是一系列基于 Makefile 规格编写的 mk 文件。主要的功能是：

1. 检测当前的编译环境是否满足 Tina Linux 的构建需求
2. 生成 host 包编译规则
3. 生成工具链的编译规则
4. 生成 target 包的编译规则
5. 生成 linux kernel 的编译规则
6. 生成系统固件的生成规则

```
build/
    ├── autotools.mk
    ├── aw-upgrade.mk
    ├── board.mk
    ├── cmake.mk
    ├── config.mk
    ├── debug.mk
    ├── depends.mk
    ├── device.mk
    ├── device_table.txt
    ├── download.mk
    ├── dumpvar.mk
    ├── envsetup.sh
    .....
```

5.3.2 config 目录

config 目录主要存放 Tina Linux 中配置菜单的界面以及一些固定的配置项，该配置菜单基于内核的 mconf 规格书写。

```
config/
    ├── Config-build.in
    ├── Config-devel.in
    ├── Config-images.in
    ├── Config-kernel.in
    └── top_config.in
```

5.3.3 docs 目录

docs 目录主要存放用于开发的文档，以 markdown 格式书写。

```
docs/
    ├── build.md
    ├── config.md
    ├── init-scripts.md
    ├── Makefile
    ├── network.md
    ├── tina.md
    ├── wireless.md
    └── working.md
```

5.3.4 lichee 目录

lichee 目录主要存放 uboot 和内核代码。

```
lichee/
    ├── bootloader
    │   ├── uboot_2011_sunxi_spl
    │   └── uboot_2014_sunxi_spl
    ├── brandy
    ├── brandy-2.0
    ├── linux-3.4
    ├── linux-3.10
    ├── linux-4.4
    └── linux-4.9
```

5.3.5 package 目录

package 目录存放 target 机器上的软件包源码和编译规则，目录按照目标软件包的功能进行分类。

```
package/
    ├── allwinner
    ├── base-files
```

```
├── devel
├── dragonst
├── firmware
├── kernel
├── .....
└── utils
```

5.3.6 prebuilt 目录

prebuild 目录存放预编译交叉编译器，目录结构如下。

```
prebuilt/
└── gcc
    └── linux-x86
        ├── aarch64
        │   ├── toolchain-sunxi-musl
        │   └── toolchain-sunxi-glibc
        ├── arm
        │   ├── toolchain-sunxi-arm9-glibc
        │   ├── toolchain-sunxi-arm9-musl
        │   ├── toolchain-sunxi-glibc
        │   ├── toolchain-sunxi-musl
        └── host
            └── host-toolchain.txt
```

5.3.7 scripts 目录

scripts 目录用于存放 pc 端或小机端使用的一些脚本。

```
scripts/
├── arm-magic.sh
├── brcmImage.pl
├── bundle-libraries.sh
├── checkpatch.pl
├── clang-gcc-wrapper
├── cleanfile
├── clean-package.sh
├── cleanpatch
```

```
├── ......
```

5.3.8 target 目录

target 目录用于存放目标板相关的配置以及 sdk 和 toolchain 生成的规格。

```
target/
    ├── allwinner
    ├── Config.in
    ├── imagebuilder
    ├── Makefile
    ├── sdk
    └── toolchain
```

5.3.9 toolchain 目录

toolchain 目录包含交叉工具链构建配置、规则

```
toolchain/
    ├── binutils
    ├── Config.in
    ├── fortify-headers
    ├── gcc
    ├── gdb
    ├── glibc
    ├── info.mk
    ├── insight
    ├── kernel-headers
    ├── Makefile
    ├── musl
    └── wrapper
```

5.3.10 tools 目录

tools 目录用于存放 host 端工具的编译规则。

```
tools/
    ├── autoconf
    ├── automake
    ├── aw_tools
    ├── b43-tools
    ├── ......
```

5.3.11 out 目录

out 目录 textcolor{blue}{用于保存编译相关的临时文件和最终镜像文件}，编译后自动生成此目录，例如编译方案 banjo-dh

```
out/
    ├── banjo-dh
    └── host
```

其中 host 目录用于存放 host 端的工具以及一些开发相关的文件。banjo-dh 目录为方案对应的目录。方案目录下的结构如下

```
out/banjo-dh/
        ├── banjo-dh-uImage
        ├── banjo-dh-uImage-initramfs
        ├── banjo-dh-zImage-initramfs
        ├── boot
        ├── boot.img
        ├── compile_dir
        ├── image
        ├── md5sums
        ├── packages
        ├── rootfs.img
        ├── sha256sums
        ├── staging_dir
        ├── tina_banjo-dh_card0.img
        ├── tina_banjo-dh_uart0.img
        └── uImage.org
```

其中，

- boot.img 为最终烧写到系统 boot 分区的数据，该分区默认为 vfat 格式。

- rootfs.img 为最终烧写到系统 rootfs 分区的数据，该分区默认为 squashfs 格式。
- banjo-dh-uImage 为内核最终的镜像，会打包到 boot.img 中。
- compile_dir 为 sdk 编译 host，target 和 toolchain 的临时文件目录，存有各个软件包的源码。
- staging_dir 为 sdk 编译过程中保存各个目录结果的目录。
- packages 目录保存的是最终生成的 ipk 软件包。
- tina_banjo-dh_card0.img 为最终固件包 (系统镜像)，串口信息转递到 tf 卡座输出。
- tina_banjo-dha_uart0.img 为最终固件包 (系统镜像)，串口信息通过串口 0 输出。

5.4 SDK 更新

开发者可根据 FAE 窗口定期发布的更新说明，通过 repo sync 命令更新。

5.5 问题反馈

Allwinner 提供 AService BUG 管理系统，用来登记客户遇到的问题以及解决状态。方便双方追踪，使问题处理更加高效。后续 SDK 问题、技术问题、技术咨询等都可以提交到此 Bug 系统上，Allwinner 技术服务会及时将问题进行分发、处理和跟踪。

注：AService BUG 管理系统登录帐号需要与 Allwinner 开通确认。

6. Tina 编译打包

6.1 概述

6.2 编译系统

(1) source build/envsetup.sh
(2) lunch
(3) make [-jN]
(4) pack [-d]

其中，
步骤(1)建立编译环境，导出编译变量.
步骤(2)提示需要选择你想要编译的方案，如astar开头的为R16开发方案。
步骤(3)参数N为并行编译进程数量，依赖编译服务器CPU核心数，如4核PC，可"make -j4"
步骤(4)打包固件，-d参数使生成固件包串口信息转到tf卡座输出。
编译完成后系统镜像会打包在out/<board>/目录下

6.3 编译 boot

命令	命令有效目录	作用
mboot	tina 根目录	编译 boot0 和 uboot
mboot0	tina 根目录	编译 boot0
muboot	tina 根目录	编译 uboot

6.4 编译内核

命令	命令有效目录	作用
mkernel	tina 根目录	编译内核

6.5 重编应用

请确保进行过一次固件的编译，确保 SDK 基础已经编译，才能单独重编应用包。重编应用包应用场景一般为：只修改了应用，不想重新烧写固件，只需要安装应用安装包即可。请确保在编译前已加载 tina 环境：

```
$ source build/envsetup.sh
$ lunch
```

6.5.1 方法一

当在应用包的目录（包括其子目录）中，可执行

```
$ mm [-B]
    => B参数则先clean此应用临时文件再编译
```

示例：假设软件包路径为：tina/package/utils/rwcheck，则：

```
$ cd tina/package/utils/rwcheck
$ mm -B
```

编译出应用安装包保存路径为：

```
tina/out/<方案>/packages/base
```

6.5.2 方法二

当在 tina 的根目录，可执行：

```
$ make <应用包的路径>/clean, ==>清空应用包临时文件
$ make <应用包的路径>/install, ==>编译软件包
或者
$ make <应用包的路径>/{clean,install}, ==>先清空临时文件再编译
```

示例：假设软件包的路径为：tina/package/utils/rwcheck，则

```
$ cd tina
$ make package/utils/rwcheck/{clean,install}
```

6.6 其他命令

命令	命令有效目录	作用
make	tina 根目录	编译整个 sdk
make menuconfig	tina 根目录	启动软件包配置界面
make kernel_menuconfig	tina 根目录	启动内核配置界面
printfconfig	tina 下任意目录	打印当前 SDK 的配置
croot	tina 下任意目录	快速切换到 tina 根目录
cconfigs	tina 下任意目录	快速切换到方案的 bsp 配置目录
cdevice	tina 下任意目录	快速切换到方案配置目录
cgeneric	tina 下任意目录	快速切换到方案 generic 目录
cout	tina 下任意目录	快速切换到方案的输出目录
cgrep	tina 下任意目录	在 c / c++ / h 文件中查找字符串
minstall path/to/package/	tina 根目录	编译并安装软件包
mclean path/to/package/	tina 根目录	clean 软件包
mm [-B]	软件包目录	编译软件包,-B 指编译前先 clean

7. Tina 系统烧写

7.1 概述

本章节主要介绍如何将构建完成的镜像文件 (image) 烧写并运行在硬件设备上的流程。Tina 提供的几种镜像烧写工具介绍如表所示，用户可以选择合适的烧写方式进行烧写。

工具	运行系统	描述
PhoenixSuit	windows	分分区升级及整个固件升级工具
PhoenixSuit	windows	卡固件制作工具
PhoenixUSBpro	windows	量产升级工具, 支持 USB 一拖 8 烧录
LiveSuit	Ubuntu	分分区升级及整个固件升级工具

PhoenixSuit 的使用请参考 TinaSDK/docs/工具文档/《PhoenixSuit 使用说明书.doc》，LiveSuit 的使用请参考 TinaSDK/docs/工具文档/《LiveSuit 使用说明书.doc》。

8. Tina uboot 定制开发

8.1 概述

本章节简单介绍 uboot 基本配置、功能裁剪、编译打包、常用命令的使用，帮助客户了解 Tina 平台 uboot 框架，为 boot 定制开发提供基础。

目前 Tina SDK 共有三版 uboot，分别是 uboot-2011、uboot-2014、uboot-2018，分别在不同硬件平台上使用，客户拿到 SDK 需要根据开发的硬件平台核对内核信息。

8.2 代码路径

```
TinaSDK/
├── brandy
│   ├── ...
│   ├── u-boot-2011
│   └── u-boot-2014
├── brandy-2.0
│   ├── ...
│   └── u-boot-2018
```

8.3 uboot 功能

TinaSDK 中，bootloader/uboot 在内核运行之前运行，可以初始化硬件设备、建立内存空间映射图，从而将系统的软硬件环境带到一个合适状态，为最终调用 linux 内核准备好正确的环境。在 Tina 系统平台中，除了必须的引导系统启动功能外，uboot 还提供烧写、升级等其它功能。

- 引导内核
  能从存储介质（nand/mmc/spinor）上加载内核镜像到 DRAM 指定位置并运行。
- 量产 & 升级
  包括卡量产，USB 量产，私有数据烧录，固件升级。

- 电源管理
  包括进入充电模式时的控制逻辑和充电时的显示画面。
- 开机提示信息
  开机能显示启动 logo 图片 (BMP 格式)。
- Fastboot 功能
  实现 fastboot 的标准命令，能使用 fastboot 刷机。

8.4 uboot 配置

以 uboot-2018 为例，各项功能可以通过 defconfig 或配置菜单 menuconfig 进行开启或关闭，具体配置方法如下:

8.4.1 defconfig 方式

8.4.1.1 defconfig 配置步骤

1. vim /TinaSDK/lichee/brandy2.0/u-boot-2018/configs/sun8iw18p1_defconfig (若是spinor方案则打开sun8iw18p1_nor_defconfig)
2. 打开sun8iw18p1_defconfig或sun8iw18p1_nor_defconfig后，在相应的宏定义前去掉或添加"#"即可将相应功能开启或关闭。

```
CONFIG_SPINOR_UBOOT_OFFSET=128
CONFIG_SPINOR_LOGICAL_OFFSET=2016

# flash
CONFIG_SUNXI_FLASH=y
CONFIG_SUNXI_NAND=y
CONFIG_SUNXI_SPINOR=y
```

图 9: defconfig 配置图

如上图, 只要将 CONFIG_SUNXI_NAND 前的 # 去掉即可支持 NAND 相关功能，其他宏定义的开启关闭也类似。

8.4.1.2 defconfig 配置宏介绍

如下图是 sun8iw18p1_defconfig/sun8iw18p1_nor_defconfig 中的基本宏定义的介绍：

宏定义	宏定义功能说明	宏定义默认状态（开启/关闭）
CONFIG_SUNXI_NAND	支持 nand 驱动的开关	（sun8iw18p1_nor_defconfig 中默认关闭，sun8iw18p1_defconfig 中默认开启）
CONFIG_SUNXI_SPINOR	支持 spinor 驱动的开关	开启
CONFIG_SUNXI_USB	支持 usb 驱动的开关	开启
CONFIG_SUNXI_EFEX	支持 usb 烧录功能的开关	开启
CONFIG_SUNXI_BURN	支持烧 key 功能的开关	开启
CONFIG_SUNXI_FASTBOOT	支持 fastboot 功能的开关	开启
CONFIG_EFI_PARTITION	支持 gpt 功能的开关	开启
CONFIG_ANDROID_BOOT_IMAGE	支持 android 固件启动功能的开关	开启
CONFIG_SUNXI_SPRITE	支持量产功能的总开关	开启
CONFIG_SUNXI_SECURE_STORAGE	支持安全存储的开关	开启
CONFIG_SUNXI_SECURE_BOOT	支持安全启动的开关	开启
CONFIG_SUNXI_KEYBOX	支持信用链的开关	开启
CONFIG_CMD_SUNXI_EFEX	shell 命令支持跳烧写命令的开关	开启
CONFIG_CMD_SUNXI_BURN	shell 命令支持烧 key 命令的开关	开启
CONFIG_CMD_GPT	shell 命令支持 gpt 命令的开关	开启
CONFIG_CMD_FAT	shell 命令支持 fat 命令的开关	开启
CONFIG_CMD_FASTBOOT	shell 命令支持 fastboot 命令的开关	开启
CONFIG_CMD_SUNXI_DMA	shell 命令支持 dma 测试命令的开关	开启
CONFIG_CMD_PART	shell 命令支持查看分区信息命令的开关	开启
CONFIG_CMD_SF	shell 命令支持 spi flash 读写命令的开关	开启

图 10: defconfig 基本宏定义介绍图

8.4.2 menuconfig 方式

通过 menuconfig 方式配置的方法步骤如下:

```
cd /TinaSDK/lichee/brandy2.0/u-boot-2018/
make ARCH=arm menuconfig 或 make ARCH=arm64 menuconfig

注意：arm针对32位平台，arm64针对64位平台
```

执行上述命令会弹出 menuconfig 配置菜单，如下图所示，此时即可对各模块功能进行配置，配置方法 menuconfig 配置菜单窗口中有说明。

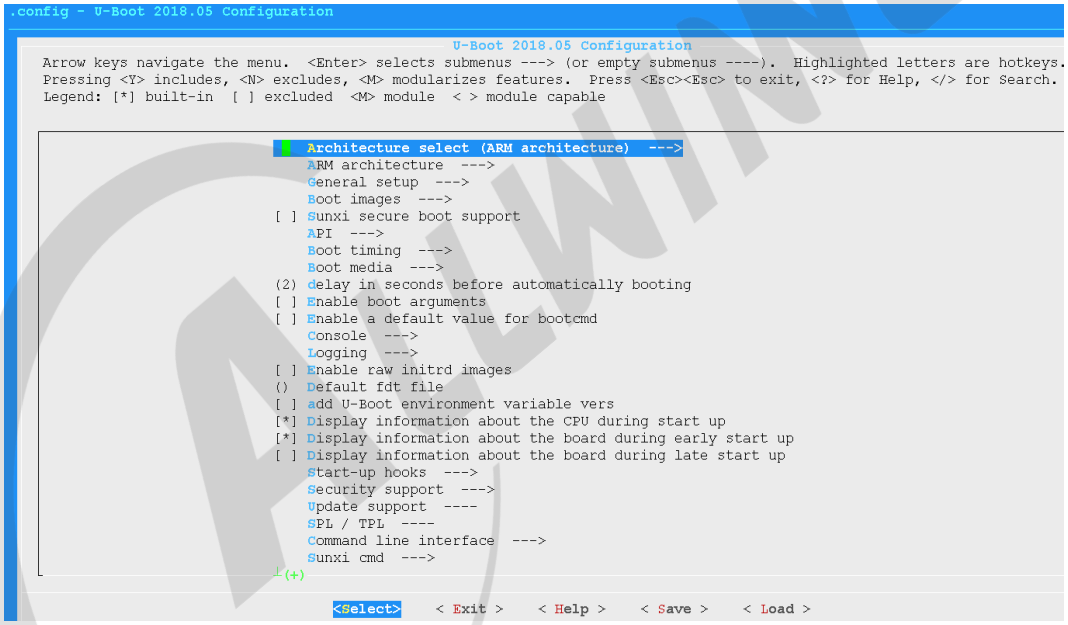

图 11: menuconfig 配置菜单图

8.5 uboot 编译

8.5.1 方法一

在 tina 目录下即可编译 uboot

```
sourcebuild/envsetup.sh(见详注1)
lunch (见详注2)
muboot(见详注3)
详注：
1加载环境变量及tina提供的命令
2输入编号，选择方案
3编译uboot,编译完成后自动更新uboot binary到TinaSDK/target/allwinner/$(BOARD)-common/bin/
```

8.5.2 方法二

```
sourcebuild/envsetup.sh(见详注1)
lunch (见详注2)
cboot(见详注3)
make XXX_config(见详注4)
make -j
详注：
3跳转到Uboot源码目录
4选择方案配置，如果是使用norflash,运行make XXX_nor_config
5执行编译uboot的动作
```

8.6 uboot 的配置

8.6.1 sys_config 配置

sys_config.fex 是对不同模块参数进行配置的重要文件，对各模块重要参数的更改及更新提供了极大的方便。其文档存放路径：/TinaSDK/target/allwinner/$(BOARD)/configs/sys_config.fex

8.6.1.1 sys_config.fex 结构介绍

sys_config.fes 主要由主键和子键构，主键是某项功能或模块的主标识，由 [] 括起，子键是对该功能或模块中各个参数的配置项，如下图所示，dram_para 是主键，dram_clk、dram_type 和 dram_zp 是子键。

```
;*****************************************************************************
;sdram configuration
;
;*****************************************************************************
[dram_para]
dram_clk            = 0
dram_type           = 3
dram_zq             = 0x000bfb
dram_odt_en         = 0x00
dram_para1          = 0x000010d2
dram_para2          = 0x0000
```

图 12: sysconfig.fex 基本结构图

8.6.1.2 sys_config.fex 配置实例

[platform]：平台相关配置项

```
[platform]
eraseflag   = 1
debug_mode = 1
next_work = 2
```

图 13: platform 配置图

例如，debug_mode =1 表示开启 uboot 的调试模式，开启后会在 log 中打印出对应的调试信息。next_work=2 表示烧录完成后系统的下一步执行动作 (0x1 表示正常启动、0x2 表示重启、0x3 表示关)，其他配置可以查看 [platform] 前的提示说明。

```
[target]
boot_clock      = 1008
storage_type    = 5
burn_key = 1
```

图 14: target 配置图

上图中的可以通过配置 boot_clock 配置 cpu 的频率大小。

```
[uart_para]
uart_debug_port = 1
uart_debug_tx   = port:PA04<3><1><default><default>
uart_debug_rx   = port:PA05<3><1><default><default>
```

图 15: uart_para 配置图

上图中的 uart_debug_port=0 表示使用的是 uart0，uart_debug_tx/uart_debug_rx 配置的 gpio 口（PA04/PA05）需要根据对应的 GPIO DATASHEET 进行配置。

8.6.1.3 sys_config.fex 解析流程

在 uboot4014/2018 中 sys_config.fex 最终会被转化为 dtb（device tree binary，linux 内核配置方式），dtb 最终会被打包烧录至 flash 中，启动过程中会将该文件加载至内存，之前在 sys_config.fex 中配置的参数已转化为 dtb 节点，最终会调用 fdt_getprop_32() 函数对 dtb 中的节点进行解析。

8.6.2 环境变量配置

uboot 的环境变量就是一个个的键值对，操作接口为：getenv()，setenv()，saveenv()。环境变量的形式：

```
boot_normal=sunxi_flash read 40007800 boot;boota 4000780\
boot_recovery=sunxi_flash read 40007800 recovery;boota 40007800\
boot_fastboot= fastboot
```

8.6.2.1 环境变量作用

可以把一些参数信息或者命令序列定义在该环境变量中。在环境变量中定义 UBOOT 命令序列，可以把 UBOOT 各个功能模块按顺序组合在一起执行，从而完成某个重要功能。

例如, 如果执行了上述提到的 boot_normal 环境变量对应的命令，Uboot 则会先调用 sunxi_flash 命令从存储介质的 boot 分区上加载内核到 DRAM 的 0x40007800 位置；然后调用 boota 命令完成内核的引导。

uboot 启动时调用环境变量方式下如图所示:

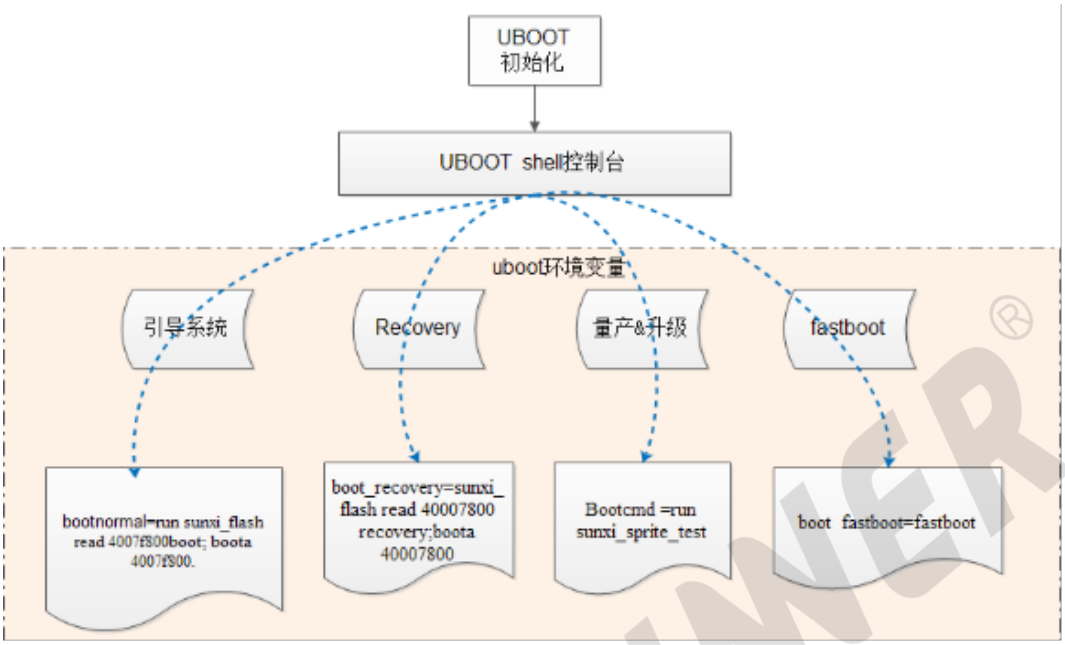

图 16: uboot 启动调用环境变量方式图

8.6.2.2 环境变量配置示例介绍

TinaSDK 中，环境变量配置文件保存在 TinaSDK/target/allwinner/$(BOARD)/configs/env.cfg 文件，用户使用的时候，可能会看到 env-4.4.cfg、env-4.9.cfg 等文件，env-xxx 后缀数字表示在不同内核版本上的配置。打开后其内容示例如下，

- bootdelay=0，改环境变量 bootdelay（即 boot 启动时 log 中的倒计时延迟时间）值的大小，为便于调试，bootdelay 的值一般不要等于 0，这样在小机上电后按下任意键才能进入 uboot shell 命令状态。
- boot_normal=sunxi_flash read 40007800 boot;boota 4000780，设置启动内核命令，即将 boot 分区读到内存 0x40007800 地址处，然后从内存 0x40007800 地址处启动内核。
- Setargs_nand=setenv bootargs earlyprintk=${earlyprink}……，设置内核相关环境变量，该变量在启动至内核的 log 中会打印处理，即 cmdline 如下图：

```
[    0.000000] Kernel command line: earlyprintk=sunxi-uart,0x05000c00 initcall_debug=0 console=ttyS0,115200 log
level=8 root=/dev/mmcblk0p7 rootwait init=/init partitions=boot-resource@mmcblk0p2:env@mmcblk0p5:boot@mmcblk0p6
:rootfs@mmcblk0p7:UDISK@mmcblk0p1 cma=256M androidboot.mode=normal androidboot.hardware=sun50iw6p1 boot_type=2
```

图 17: kernel cmdline 图

- loglevel=8，设置内核 log 打印等级。

8.6.3 sys_partition.fex 分区配置

分区配置文件是一个规划磁盘分区的文件，烧录过程会按照该分区配置文件将各分区数据烧录至 flash 中。

TinaSDK 中，分区配置文件路径 TinaSDK/target/allwinner/$(BOARD)/configs/sys_partition.fex。有些方案可以看到 sys_partition.fex、sys_partition_nor.fex 两个分区配置文件，若是打包 Tina 非 nor 固件，则使用的是 sys_partition_linux.fex 配置文件，若是打包 nor 固件, 则使用的是 sys_partition_nor.fex。

8.6.3.1 sys_partition.fex 分区配置介绍

一个分区的属性，包含名称、分区大小、下载文件与用户属性。以下是文件中所描述的一个分区的属性：

- name，分区名称由用户自定义。当用户在定义一个分区的时候，可以把这里改成自己希望的字符串, 但是长度不能超过 16 个字节
- size，定义该分区的大小，以扇区的单位 (1 扇区 =512bytes, 如上图给 env 分区分配了 32768 个扇区，即 32768*512/1024/1024 = 16M), 注意，为了字节对齐，这里分配的扇区大小应当能整除 128。
- downloadfile，下载文件的路径和名称。可以使用相对路径，相对是指相对于 image.cfg 文件所在分区。也可以使用绝对路径。
- user_type，提供给操作系统使用的属性。目前，每个操作系统在读取分区的时候，会根据用户属性来判断当前分区是不是属于自己的然后才进行操作。这样设计的目的是为了避免在多系统同时存在的时候，A 操作系统把 B 操作系统的系统分区进行了不应该的读写操作，导致 B 操作系统无法正常工作。

9. Tina kernel 定制开发

9.1 概述

本章节简单介绍 kernel 基本配置、功能裁剪、常用命令的使用，帮助客户了解 Tina 平台 linux 内核，为内核定制开发提供基础。

目前 Tina SDK 共有 4 版 linux kernel，分别是 linux-3.4、linux-3.10、linux-4.4、linux-4.9，分别在不同硬件平台上使用，客户拿到 SDK 需要根据开发的硬件平台核对内核信息。

9.2 代码路径

```
TinaSDK/
    ├── ...
    ├── linux-3.10
    ├── linux-3.4
    ├── linux-4.4
    └── linux-4.9
```

9.3 模块开发文档

详阅 BSP 开发文档，文档目录包括常用内核模块使用与开发说明。

9.4 内核配置

客户在定制化产品时，通常需要更改 linux 内核配置，在 TinaSDK 中，打开内核配置的方式如下，

```
croot
make kernel_menuconfig
```

执行完后，shell 控制台会跳出配置菜单。如下图所示，

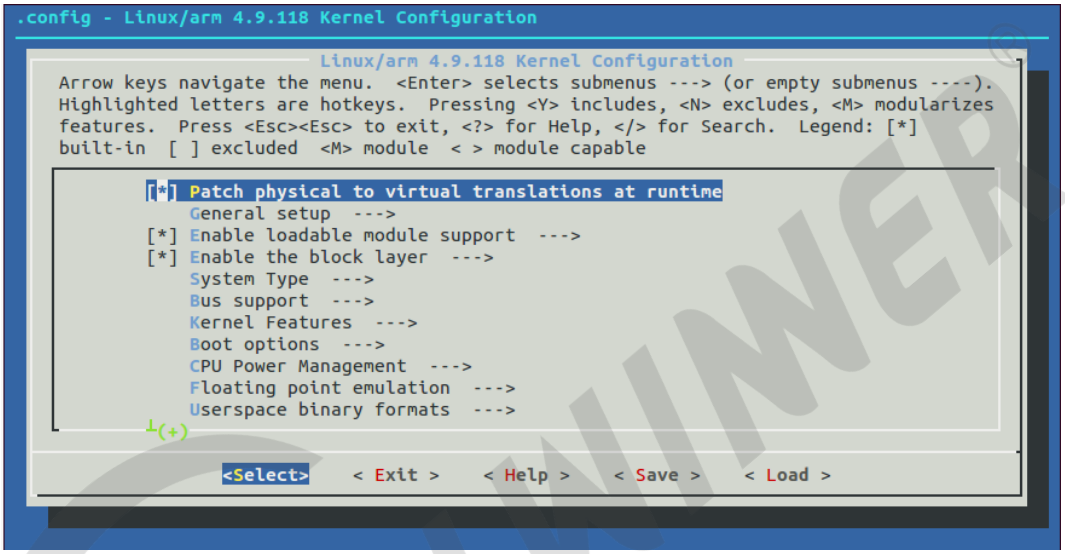

图 18: TinaLinux 内核配置菜单

10. Tina 系统定制开发

10.1 应用移植

在 Tina Linux SDK 中一个软件包目录下通常包含如下两个目录和一个文件:

```
package/<分类>/<软件包名>/Makefile
package/<分类>/<软件包名>/patches/ [可选]
package/<分类>/<软件包名>/files/ [可选]

其中，
patches 保存补丁文件，在编译前会自动给源码打上所有补丁
files 保存软件包的源码，在编译时会对应源码覆盖源码中的源文件
Makefile 编译规则文件，
```

10.1.1 Makefile 范例

该 Makefile 的功能是软件源码的准备, 编译和安装的过程, 提供给 Tina Linux 识别和管理软件包的接口, 软件的编译逻辑是由软件自身的 Makefile 决定, 理论上和该 Makefile(该 Makefile 只执行 make 命令和相关参数) 无实质关系.

```
include $(TOPDIR)/rules.mk

PKG_NAME:=bridge                                                    #软件包的名字
PKG_VERSION:=1.0.6                                                  #软件包版本(详见①)
PKG_RELEASE:=1                                                      #该Makefile的版本

PKG_SOURCE:=bridge-utils-$(PKG_VERSION).tar.gz                      #软件包文件名(dl目录下)
PKG_SOURCE_URL:=@SF/bridge                                          #该包的下载地址(详见②)
PKG_MD5SUM:=9b7dc52656f5cbec846a7ba3299f73bd                        #该包的md5值(详见③)
PKG_CAT:=zcat                                                       #该软件包的解压方法
PKG_BUILD_DIR:=$(COMPILE_DIR)/bridge-utils-$(PKG_VERSION)           #该软件包的编译目录

include $(BUILD_DIR)/package.mk

define Package/bridge                                  #Package/<包名>(详见④)
   SECTION:=net                                        #该包的类别,目前没有使用
   CATEGORY:=Base system                               #该包在配置系统中的分类
   TITLE:=Ethernet bridging configuration utility      #该包的简单的描述信息
   URL:=http://bridge.sourceforge.net/                 #该包的原始下载连接
   [可选]MAINTAINER:=                                  #该包的维护人员的联系方式
   [可选]DEPENDS:=                                     #该包的依赖关系(详见⑤)
   [可选]BUILDONLY:=                                   #固定编译(详见⑥)
endef

[可选]
define Package/<name>/conffiles:
        #指定软件包依赖的配置文件,一个配置文件一行
endef

[可选]
define Build/Prepare:
        #准备源码和给源码打补丁
endef

[可选]
define Package/bridge/description
        #该软件包的文字描述
endef

[可选]
define Build/Configure
        #配置软件包(详见⑦)
)
endef

[可选]
define Build/Compile
        #执行软件编译的动作(详见⑧)
endef

Define Package/bridge/install                          #执行软件包的安装动作(详见⑨)
   $(INSTALL_DIR) $(1)/usr/sbin
   $(INSTALL_BIN) $(PKG_BUILD_DIR)/brctl/brctl $(1)/usr/sbin/
endef

$(eval $(call BuildPackage,bridge))                    #BuildPackage的宏(详见⑩)
```

图 19: Tina Package Makefile 示例

详注：

```
1.如果是开源软件,软件包版本建议与下载软件包的版本一致
2.以PKG开头的变量主要告诉编译系统去哪里下载软件包.
3.md5sum用于校验下载下来的软件包是否正确,如果正确,在编译该软件的时候,就会在PKG_BUILD_DIR下找到该软件包的源码.
4.Package/<name>: <name>用来指定该Package的名字,该名字会在配置系统中显示
5.使用依赖包的名字<name>来指定依赖关系,如果是扩展包,前面添加一个”+”号,如果是内核版本依赖使用@LINUX_2_<minor version>
6.如果该值为1,该包将不会出现在配置菜单中,但会作为固定编译,可选
7.在开源软件中一般用来生成Makefile,其中参数可以通过CONFIGURE_VARS来传递
8.在开源软件中一般相当于执行make,其中有两个参数可以使用:MAKE_FLAGS和MAKE_VARS
9.内置的几个关键字如下:
    INSTALL_DIR相当于install -d m0755
    INSTALL_BIN相当于install -m0755
    INSTALL_DATA相当于install -m0644
    INSTALL_CONF相当于install -m0600
10.该Makefile的所有define部分都是为该宏的参数做的定义.上层Makefile通过调用此宏进行编译
```

10.1.2 自启动设置

在 Tina Linux 中支持两种格式的初始化脚本，一种是 busybox 式或者 sysV 式的初始化脚本，一种是 procd 式的初始化脚本。一般我们把由初始化脚本启动的应用叫做服务。

初始化脚本以 shell 脚本的编程语言组织，shell 脚本作为基础知识在此不展开说明。一般情况下，初始化脚本源码保存在软件的 files 目录，且后缀为 ``.init"，例如

```
tina/package/system/fstools/files/fstab.init
```

在 Makefile 的 install 中把初始化脚本安装到小机端的etcinit.d 中，例如：

```
define Package/block-mount/install
    $(INSTALL_DIR) $(1)/etc/init.d/
    $(INSTALL_BIN) ./files/fstab.init $(1)/etc/init.d/fstab
endef
```

10.1.2.1 调用自启动脚本

- 手动调用方式在启动的时候会有太多的 log，且 log 信息已被 logd 守护进程收集，不利于我们调试初始化脚本，此时可通过小机端的命令行手动调用的形式来调试，例如：

```
root@TinaLinux: /# /etc/init.d/fstab start
```

10.1.2.2 sysV 格式脚本

sysV 式的初始化脚本保存在小机端的/etc/init.d/目录下，实现开机自启动。下例以最小内容的初始化脚本作示例讲解，核心是实现 start/stop 函数:

```
#!/bin/sh /etc/rc.common
# Example script
# Copyright (C) 2007 OpenWrt.org

START=10
STOP=15
DEPEND=xxxx

start() {
    #commands to launch application
}
stop() {
    #commands to kill application
}

注意:
START=10， 指明开机启动优先级(序列) [数值越小, 越先启动]，取值范围0-99
STOP=15， 指明关机停止优先级(序列) [数值越小, 越先关闭]，取值范围0-99
DEPEND=xxxx， 指明初始化脚本会并行执行，通过此项配置确保执行的依赖
```

在 rc.common 中提供了一个 init 脚本的功能模板，模板中包括如下几个组成部分：

名称	属性	功能
start	必须实现	启动一个服务
stop	必须实现	停止一个服务
reload	可选实现	重启一个服务
enable	可选实现	重新加载服务
disable	可选实现	禁用服务

在 shell 里面可以使用如下的命令来操作相关的服务。

```
$ root@TinaLinux:/# /etc/init.d/exmple restart|start|stop|reload|enable|disable
```

10.1.2.3 procd 格式脚本

以下例的初始化脚本作示例讲解，主要是实现函数 start_service:

```
#!/bin/sh /etc/rc.common
USE_PROCD=1
PROG=xxxx
START=10
STOP=15
DEPEND=xxxx

start_service() {
        procd_open_instance
        procd_set_param command $PROG -f
        ......
        procd_close_instance
}
```

详细的介绍可以参考：https://wiki.openwrt.org/inbox/procd-init-scripts

10.2 应用调试

新添加的软件默认配置为不使能，此时需要手动配置使能软件包。通过在 tina 的根目录执行 make menuconfig 进入软件包的配置界面：

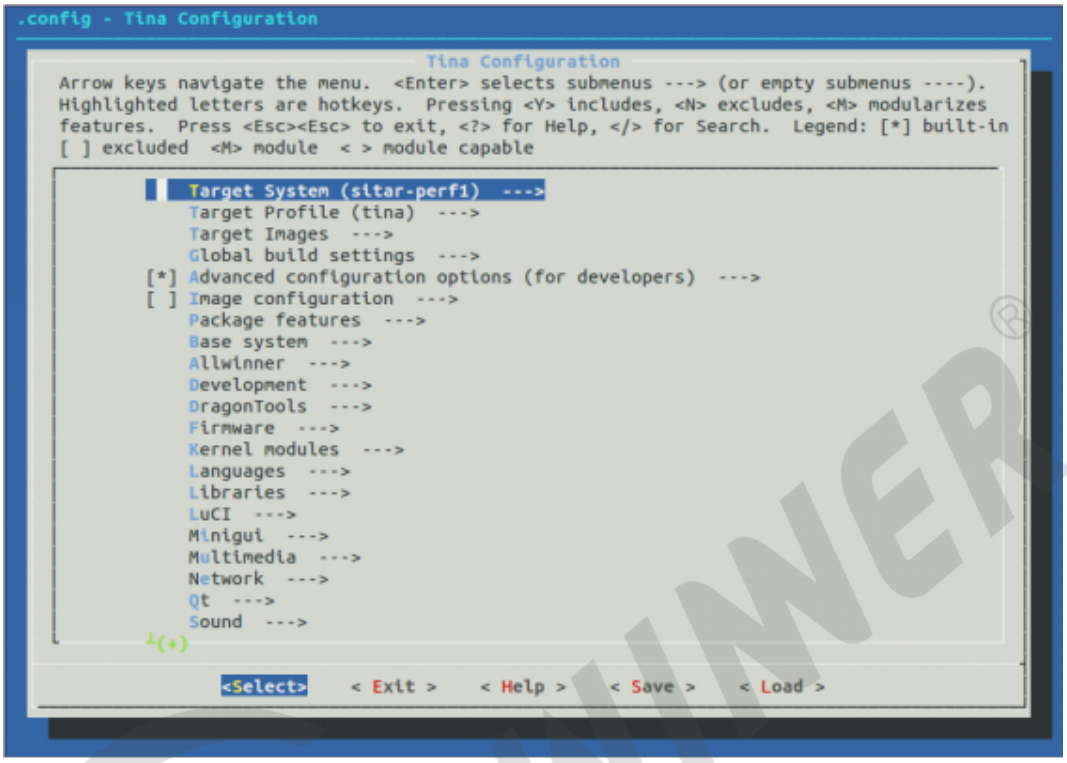

图 20: 应用配置主界面

软件包的所在路径与软件包的 Makefile 中的定义有关，以 fstools 为例，在 Makefile 中定义为：

```
define Package/fstools
  SECTION:=base
  CATEGORY:=Base system
  DEPENDS:=+ubox +USE_GLIBC:librt +NAND_SUPPORT:ubi-utils
  TITLE:=OpenWrt filesystem tools
  MENU:=1
endef
```

此时，只需要在 menuconfig 界面中进入 Basy system 即可找到 fstools 的软件包

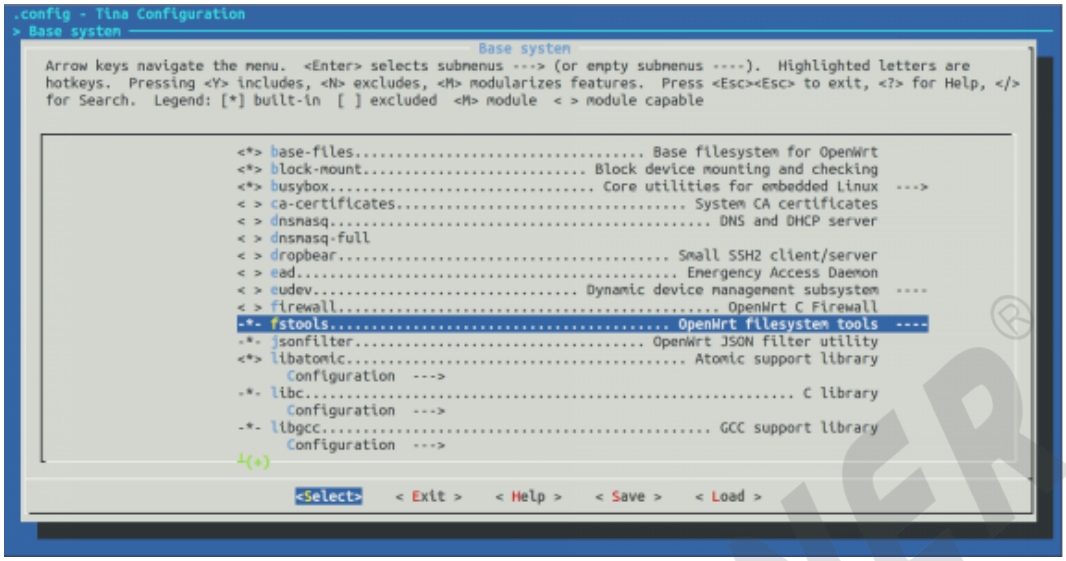

图 21: 软件包所在界面

前缀符号含义:

[*] 或 <*>： 编译进入SDK
[ ] 或 < >： 不包含

支持操作：

Y 或 y：选择包含
N 或 n：取消选择

10.3 应用编译

详见 6.5 节重编应用。

10.4 应用编译

1. 获取安装包

安装包一般位于目录：

```
tina/out/<方案>/packages/base
```

安装包命名格式为：

```
<应用名>_<应用版本>-<应用释放版本>_sunxi.ipk
```

2. 安装应用包

通过 adb 推送安装包到小机

```
$ adb push <安装包路径> <推送到小机路径>
```

安装应用包

```
$ opkg install <安装包路径>
```

10.5 分区与挂载

- 升级分区

分区	功能
boot 分区	存内核镜像
rootfs 分区	基础系统镜像分区，包含 (/lib，/bin，/etc 等)
recovery 分区	存放恢复系统镜像 [仅大容量方案有]

- 不升级分区

分区	功能
private 分区	存储 SN 号分区
misc 分区	系统状态、刷机状态分区
UDISK 分区	用户数据分区（/mnt/UDISK）
overlayfs 分区	存储 overlayfs 覆盖数据

- 默认挂载点

分区	挂载点	备注
/dev/by-name/boot	/boot	
/dev/by-name/boot-res	/boot-res	
/dev/by-name/UDISK	/mnt/UDISK	用户数据分区
/dev/mmcblk0 或 /dev/mmcblk0p1	/mnt/SDCARD	Tf 卡挂载点
/dev/by-name/rootfs_data	/overlay	存储 overlayfs 覆盖数据

11. Declaration

This document is the original work and copyrighted property of Allwinner Technology (“Allwinner”). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner. The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This datasheet neither states nor implies warranty of any kind, including fitness for any particular application.tates nor implies warranty of any kind, including fitness for any particular application.