



D1-H Tina Linux 系统裁剪 开发指南

版本号: 1.0
发布日期: 2021.04.20

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.04.20	AWA0916	first version



目 录

1 概述	1
1.1 编写目的	1
1.2 适用范围	1
1.3 相关人员	1
2 Tina 系统裁剪简介	2
2.1 boot0 裁剪	2
2.2 uboot 裁剪	2
2.3 内核裁剪	2
2.3.1 删除不使用的功能	3
2.3.2 删除不使用的驱动	4
2.3.3 修改内核源代码	4
2.3.3.1 size 工具	4
2.3.3.2 ksize.py 脚本	4
2.3.3.3 nm 命令	8
2.4 文件系统裁剪	9
2.4.1 应用程序及冗余文件裁剪	9
2.4.2 库的裁剪	9
2.4.2.1 C 库的选择	10
2.4.2.2 删除没用到的库	10
2.4.3 应用程序与库 strip	10
2.4.4 文件系统压缩	11
3 参考资料	12

1 概述

1.1 编写目的

嵌入式产品往往为了压缩成本而使用较小的 flash 存储器，因此可能需要对系统进行裁剪来减少对 flash 的占用。系统经过裁剪过后，通常也会提升启动速度以及减少内存占用。

本文介绍 TinaLinux 中系统裁剪的方法，为有裁剪需求的使用者提供参考。

1.2 适用范围

适用于硬件平台：全志 D1-H 芯片。

软件平台：Tina V3.5 及其后续版本。

1.3 相关人员

适用于 TinaLinux 平台的客户及相关技术人员。

2 Tina 系统裁剪简介

Tina 固件中通常包含 boot0、uboot、kernel、rootfs 等镜像。基于经验，各个镜像尺寸的量级如下表所示：

表 2-1: 各镜像尺寸的量级

镜像	大小
boot0	< 100K
uboot	< 1M
kernel	$\geq 3\text{M}$, < 15M
rootfs	$\geq 4\text{M}$

可以看到 boot0、uboot、kernel、rootfs 的尺寸是依次增大的。对于大尺寸的裁剪效果往往比小尺寸的裁剪效果明显，比如 rootfs 裁剪 1M 可能很容易，对于 uboot 来说，则非常困难。

因此，后续主要介绍 kernel 以及 rootfs 的裁剪。

2.1 boot0 裁剪

由于 boot0 很小，因此略过。

2.2 uboot 裁剪

uboot 代码位于 tina/lichee/brandy*/u-boot* 目录下，主要有下面两种裁剪思路：

- 修改 uboot 配置文件，删减不需要的配置。uboot 配置文件通常位于源码下 include/configs/\${CHIP}.h 或者 configs/\${CHIP}*_defconfig。
- 删除不需要的 uboot 命令。

2.3 内核裁剪

通常关于 Linux 内核裁剪主要有如下方法：

- 删除不使用的功能。如符号表、打印、调试等功能。
- 删除不使用的驱动。
- 修改内核源代码。
- 内核压缩（D1-H 当前不支持内核压缩）。

2.3.1 删除不使用的功能

下表中列出了一些内核选项，包含选项的描述，默认值以及推荐值（减小内核镜像尺寸）。

表 2-2: 内核选项及描述

CONFIG option	Description	Def	Small
CORE_SMALL	tune some kernel data sizes	N	Y
NET_SMALL	tune some net-related data sizes	N	Y
KMALLOC_ACCOUNTING	turn on kmalloc accounting	N	Y *
AUDIT_BOOTMEM	print out all bootmem allocations	N	Y *
DEPRECATE_INLINES	cause compiler to emit info about inlines	N	Y *
PRINTK	printk code and message data	Y	N
BUG	allow elimination of BUG code	Y	N
ELF_CORE	allow disabling of ELF core dumps	Y	N
PROC_KCORE	allow disabling of /proc/kcore	Y	N
AIO	allow disabling of async IO syscalls	Y	N
XATTR	allow disabling of xattr syscalls	Y	N
FILE_LOCKING	allow disabling of file locking syscalls	Y	N
DIRECTIO	allow disabling of direct IO support	Y	N
MAX_SWAPFILES_SHIFT	number of swapfiles	5	0
NR_LDISCS	number of tty line disciplines	16	2
MAX_USER_RT_PRIO	number of RT priority levels	100	5
KALLSYMS	load all symbols for debugging/kksymoops	Y	N
SHMEM	allow disabling of shmem filesystem	Y	N +
SWAP	support for a swap segment	Y	N
SYSV_IPC	support for System V IPC	Y	N +
POSIX_MQUEUE	POSIX message queue support	Y	N +
SYSCTL	allow disabling of sysctl support	Y	N +
LOG_BUF_SHIFT	control size of kernel printk buffer	14	11
CC_OPTIMIZE_FOR_SIZE	Use gcc -os to optimize for size	Y	Y
MODULES	allow support for kernel loadable modules	Y	N +
KMOD	automatic kernel module loading	Y	N
PCI	allow support for PCI bus and devices	Y	Y -
XIP_KERNEL	allow support for kernel Execute-in-Place	N	N
BLK_DEV_LOOP	support for loopback block device	Y	Y -
IOSCHED_AS	Include Anticipatory IO scheduler	Y	Y

CONFIG option	Description	Def	Small
IOSCHED_DEADLINE	Include Deadline IO scheduler	Y	N +
IOSCHED_CFQ	Include CFQ IO scheduler	Y	N +
IP_PNP	support for IP autoconfiguration	Y	N +
IP_PNP_DHCP	support for IP autoconfiguration via DHCP	Y	N +
IDE	support for IDE devices	Y	N +
SCSI	support for SCSI devices	Y	N +

其中：

- "Y *" - 表示开发的时候设置成 Y，发布的时候可以设置成 N。
- "N +" - 表示基于应用需要来判断是否设置成 N。
- "Y -" - 表示可能需要，可以设置 N 尝试一下。

2.3.2 删除不使用的驱动

方案明确之后，所需的内核驱动也明确了。可以执行 `make kernel_menuconfig`，将没有用到的驱动关闭。

2.3.3 修改内核源代码

内核源码庞大，直接修改往往难度很大，可借助相关工具来评估模块以及符号的大小，然后进行针对性的裁剪。

2.3.3.1 size 工具

`size` 命令可查看内核镜像的 `text`、`data`、`bss` 等段的大小。如执行 `"size vmlinux"`，将会得到：

```
text      data      bss      dec      hex filename
5818117 1378944 168972 7366033 706591 vmlinux
```

2.3.3.2 ksize.py 脚本

在 `tina/lichee/linux-5.4/scripts` 目录下有一个 `ksize` 脚本，可以对内核目录下的 `built-in.o` 进行解析，并将解析的内容按照尺寸进行排序，显示出来。执行结果如下所示：

```
xxx@xxx:~/tina/lichee/linux-5.4$ ./scripts/ksize
```

Linux Kernel (vmlinux)	total	text	data	bss
vmlinux	9403884	6664284	2482236	257364
drivers	3592850	3079263	417716	95871
net	1847651	1738679	68309	40663
fs	1317643	1271078	24055	22510
kernel	624683	509903	65823	48957
sound	489421	453897	30259	5265
lib	438875	430156	6418	2301
mm	367697	339419	26018	2260
crypto	201590	173607	18584	9399
block	154302	145789	6069	2444
arch/riscv	76482	25118	24524	26840
security	39493	37513	1920	60
ipc	33016	31226	1790	0
init	20222	11922	8176	124
certs	1202	1178	16	8
sum	9205127	8248748	699677	256702
delta	198757	-1584464	1782559	662

drivers	total	text	data	bss
drivers/built-in.a	3592850	3079263	417716	95871
drivers/video	609591	443517	121892	44182
drivers/usb	536603	463662	51249	21692
drivers/media	328722	271446	54350	2926
drivers/tty	196446	165943	19922	10581
drivers/mtd	182383	171037	8226	3120
drivers/base	181696	169126	11447	1123
drivers/mmc	161588	156627	4745	216
drivers/scsi	116639	105241	10732	666
drivers/clk	116554	82738	33740	76
drivers/hid	104678	96134	8464	80
drivers/net	86845	83547	3262	36
drivers/input	72865	69090	3548	227
drivers/pinctrl	71252	48352	22872	28
drivers/char	68430	61621	5657	1152
drivers/of	61556	57273	619	3664
drivers/regulator	60033	55577	3896	560
drivers/spi	58207	54422	3760	25
drivers/cpufreq	55287	51479	3720	88
drivers/i2c	53655	51989	1630	36
drivers/iommu	43846	42510	1232	104
drivers/gpio	43045	42137	900	8
drivers/crypto	37724	21351	16365	8
drivers/leds	33385	32480	881	24
drivers/rtc	24432	23430	928	74
drivers/pwm	23234	21954	1064	216
drivers/dma	22973	21507	1342	124
drivers/power	22262	14382	7176	704
drivers/bluetooth	22085	21186	498	401
drivers/opp	21708	21440	260	8
drivers/dma-buf	21474	20770	576	128
drivers/misc	19542	18270	1232	40
drivers/staging	17456	13695	801	2960

drivers/nvmm	13541	12429	1112	0
drivers/virtio	13495	12742	753	0
drivers/cpuidle	11847	9855	1932	60
drivers/watchdog	11217	10550	633	34
drivers/rpmsg	9390	8414	972	4
drivers/reset	8574	8058	516	0
drivers/bus	8506	6758	1668	80
drivers/hwmon	7114	6854	260	0
drivers/clocksource	6086	5154	904	28
drivers/mfd	5454	5130	324	0
drivers/soc	4523	4039	112	372
drivers/irqchip	3392	2884	492	16
drivers/pci	131	131	0	0

sum	3579466	3066931	416664	95871
delta	13384	12332	1052	0

net	total	text	data	bss
-----	-------	------	------	-----

net/built-in.a	1847651	1738679	68309	40663
----------------	---------	---------	-------	-------

net/ipv4	434939	402244	20499	12196
net/core	363444	341957	16210	5277
net/ipv6	331413	306767	11586	13060
net/bluetooth	292219	289286	2712	221
net/wireless	202989	193930	6651	2408
net/netfilter	63608	57866	3176	2566
net/can	33114	31431	1571	112
net/packet	28414	27602	810	2
net/unix	27370	22377	880	4113
net/netlink	27065	25721	1316	28
net/*.o	16859	15858	529	472
net/sched	12285	11036	1249	0
net/rfkill	10110	8874	1028	208
net/ethernet	3822	3730	92	0

sum	1847651	1738679	68309	40663
delta	0	0	0	0

fs	total	text	data	bss
----	-------	------	------	-----

fs/built-in.a	1317643	1271078	24055	22510
---------------	---------	---------	-------	-------

fs/*.o	392348	368941	6791	16616
fs/ext4	303025	295047	7134	844
fs/ubifs	221876	221016	740	120
fs/proc	78413	77342	999	72
fs/overlayfs	63880	62119	1744	17
fs/fat	52900	52288	564	48
fs/jbd2	50771	48650	2001	120
fs/debugfs	26161	25913	228	20
fs/kernfs	23485	18979	394	4112
fs/squashfs	20737	20601	128	8
fs/configfs	20502	19698	780	24
fs/notify	16497	15144	868	485
fs/nls	14585	14353	232	0
fs/iomap	14526	14178	348	0
fs/sysfs	9087	8779	292	16

fs/devpts	3925		3249	668	8
fs/ramfs	2643		2523	120	0
fs/exportfs	2282		2258	24	0

sum	1317643		1271078	24055	22510
delta	0		0	0	0

kernel	total		text	data	bss

kernel/built-in.a	624683		509903	65823	48957

kernel/*.o	262962		224298	26543	12121
kernel/time	100055		78975	15364	5716
kernel/printk	55146		18429	8649	28068
kernel/sched	51083		43736	7147	200
kernel/rcu	45836		42303	3400	133
kernel/irq	40744		36488	2132	2124
kernel/locking	20821		20141	676	4
kernel/power	19949		17860	1560	529
kernel/bpf	16404		16228	140	36
kernel/dma	11683		11445	212	26

sum	624683		509903	65823	48957
delta	0		0	0	0

sound	total		text	data	bss

sound/built-in.a	489421		453897	30259	5265

sound/soc	190667		179649	10174	844
sound/usb	178075		159017	18153	905
sound/core	120132		114716	1908	3508
sound/*.o	547		515	24	8

sum	489421		453897	30259	5265
delta	0		0	0	0

lib	total		text	data	bss

lib/built-in.a	438875		430156	6418	2301

lib/*.o	250335		243133	7009	193
lib/zstd	205846		205846	0	0
lib/crypto	34095		33583	512	0
lib/zlib_deflate	16526		14130	108	2288
lib/zlib_inflate	11265		11265	0	0
lib/mpi	11230		11206	24	0
lib/xz	10117		10045	72	0
lib/lzo	3951		3951	0	0
lib/math	1646		1634	12	0

sum	545011		534793	7737	2481
delta	-106136		-104637	-1319	-180

crypto	total		text	data	bss

crypto/built-in.a	201590		173607	18584	9399
crypto/*.o	182630		155152	18104	9374
crypto/asymmetric_keys	18960		18455	480	25
sum	201590		173607	18584	9399
delta	0		0	0	0
block	total		text	data	bss
block/built-in.a	154302		145789	6069	2444
block/*.o	146013		137532	6041	2440
block/partitions	8289		8257	28	4
sum	154302		145789	6069	2444
delta	0		0	0	0
arch/riscv	total		text	data	bss
arch/riscv/built-in.a	76482		25118	24524	26840
arch/riscv/mm	45772		8380	12680	24712
arch/riscv/kernel	30710		16738	11844	2128
sum	76482		25118	24524	26840
delta	0		0	0	0
security	total		text	data	bss
security/built-in.a	39493		37513	1920	60
security/keys	34655		32751	1856	48
security/*.o	4838		4762	64	12
sum	39493		37513	1920	60
delta	0		0	0	0

可以对各个模块的代码段数据段的统计信息进行确认，对占用空间大的进行针对性优化。

2.3.3.3 nm 命令

nm 命令可查看内核模块中各个符号的尺寸。如执行"nm --size -r vmlinux | head -10"，可得到：

```
000000000004000 b __log_buf
000000000003d50 d LCM_LT080B21BA94_setting
0000000000003cc0 d lcm_initialization_setting
00000000000039c0 R v4l2_dv_timings_presets
00000000000039a2 T hidinput_connect
00000000000038d0 d lcm_initialization_setting
0000000000003888 d lcm_initialization_setting
0000000000003768 d lcm_initialization_setting
0000000000002fa8 t whitelist
```

```
0000000000002f78 d sunxi_ss_algs
```

说明，一共有三列数据，分别表示大小、符号类型、符号名。其中符号类型：

- b/B - 符号位于 bss 段。
- t/T - 符号位于 text 段。
- d/D - 符号位于 data 段。
- r/R - 符号位于 rodata 段。

如果某些函数或者全局变量占用较大，可以进行针对性的优化。

2.4 文件系统裁剪

对于文件系统裁剪来说，主要思路是删、换、压。

- 删。删除不需要的内容。如帮助文档、没用到的库、调试程序等。
- 换。使用小尺寸的实现替换大尺寸的实现。如使用 mbedtls 库替换 openssl 库等。
- 压。使用合适的压缩算法。

2.4.1 应用程序及冗余文件裁剪

在不影响整体功能的情况下，一些应用程序或冗余文件往往可以删除：

- 调试工具。比如 tcpdump、mpstat、strace 等等。
- 性能测试工具。比如 lmbench、sysstat、tiobench 等等。
- 冗余文件。帮助文档、辅助程序、配置文件和数据模块等，又比如很多应用有相同的共能，只留其一。
- 采用具有通用功能的替代软件包。Linux 上有许多具有相似功能的软件包，可以选择其中占存储空间较小的软件包并移植到嵌入式设备上。
- 资源文件。一些音视频以及 UI 资源往往占用很大空间，如果没有用到，也需要删除。

2.4.2 库的裁剪

关于库的裁剪主要有两个思路：

- 使用较小的 C 库。
- 删除没有用到的库。

2.4.2.1 C 库的选择

下表列出了当前一些通用的 C 库及其特征。

表 2-3: 常用 C 库及其特征

C 库	环境	大小	优点	缺点
glibc	Distribution	大	强大稳定，支持最多的 cpu 架构	占用空间大
uclibc	Embedded	小	为嵌入式设计，可配置性好	不支持 libdb 与 libnss
bionic	Android	小	提供了 Android 特性的函数	不提供 libthread_db/libm
musl	Embedded	小	更小，高效静态链接，稳定	支持较少的 cpu arch

当前 Tina 环境 riscv 方案只支持 glibc 库，后续 Tina SDK 会加入其他 C 库供选择。

2.4.2.2 删除没用到的库

嵌入式产品通常应用程序有限，因此可能存在很多库不会被用到，可以进行删除。

当前 Tina 环境提供了一种删除方法，执行 make menuconfig，打开如下选项：

```
Tina Configuration
Target Images ---->
[*] downsize the root filesystem or initramfs
```

打开之后，在生成 rootfs/initramfs 之前会对其中没有用到的库进行删除。

具体可参考 scripts/reduce-rootfs-size.sh 文件，其主要思路是：

- 分析 rootfs 下的应用程序所依赖的库。
- 分析“应用程序依赖库”所依赖的库，一直递归下去，直到完全找出所有依赖的库。
- 根据上述查找结果，删除没有被依赖的库。

📖 说明

此方法有一定的限制：

- 当前只分析 `/lib`，`/usr/lib` 下的库，其他目录不会处理。
- 对于部分使用 `dlopen` 的应用程序，解析库可能会出现问題。

2.4.3 应用程序与库 strip

strip 会去掉应用程序与库的符号信息和调试信息，大大减少空间占用。

当前 Tina 环境下默认开启了 strip 功能，如果没开启，请确保开启以减少空间占用。

```
Tina Configuration
Global build settings --->
Binary stripping method (strip) ---->
```

2.4.4 文件系统压缩

有些文件系统支持压缩，有些不支持。下表列出了常见的文件系统类型：

表 2-4: 常用文件系统类型

FS	使用	压缩	读写	备注
ext2	block device	无	RW	突然断电或当机时可能导致数据丢失
ext3	block device	无	RW	向前兼容 ext3，日志式文件系统，非常成熟稳定
ext4	block device	无	RW	向前兼容 ext2 和 ext3，扩展存储限制，提升性能
btrfs	block device	有	RW	着重于容错、修复及易管理
FAT	block device	无	RW	Windows，长期使用速度变慢，不支持 >4G 文件
NTFS	block device	有	RW	Windows，基于 FAT 做若干改进，日志文件系统
Cramfs	NAND Flash	无	RO	2013 停用，使用 Squashfs
Squashfs	Raw Flash	有	RO	压缩度更高，没有大小限制
UBIFS	Raw Flash	有	RW	基于 JFFS2，Linux3.7 之后
JFFS2	Raw Flash	有	RW	mount 时间很慢，读写性能不好
YAFFS2	NAND Flash	无	RW	没有透明压缩，不在 Linux 主线

当前 Tina 环境下比较常用的是 squashfs、ext4、jffs2 三种文件系统。具体可执行 make menu-config 进行选择：

```
Tina Configuration
Target Images --->
*** Root filesystem images ***
[ ] ext4 ----
[ ] jffs2
[*] squashfs --->
```

常见的压缩有 lzop, gzip, xz 等，压缩率最高的是 xz。但是 xz 压缩解压最慢，非常影响启动速度。实际在选择压缩方式时应综合考虑。

3 参考资料

- [1] https://elinux.org/Kernel_Size_Tuning_Guide
- [2] Karim Yaghmour. Building Embedded Linux Systems [M]
- [3] Michael Opdenacker. Embedded Linux size reduction techniques
- [4] <https://tiny.wiki.kernel.org/>






著作权声明

版权所有 © 2022 珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本文档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明

、 全志科技、（不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本文档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。