执行体全量识别与精细管控

固件引导系统的研究与实践

安天 | 部门名称

目 录

01 引导系统简介

引导系统简介

- MBR
- UEFI
- UBOOT

- 支持文件系统，默认FAT32，也可以支持其他文件系统，比如NTFS，需要加载相关模块。
- 小型操作系统，加载模块为PE格式，可提供接口给上层操作系统。

嵌入式引导系统

- UBOOT，德国人的开源项目，已经成为嵌入式的实质标准。
- 由于UBOOT功能繁多，也暴露出了更多的攻击面，比如:TFTP功能使得固件在启动阶段加载外部kernel和文件系统，从而劫持整个设备。

02 不同芯片的引导及其架构和调用方法

不同芯片的引导方式

- Allwinner
- Amlogic
- Rockchip

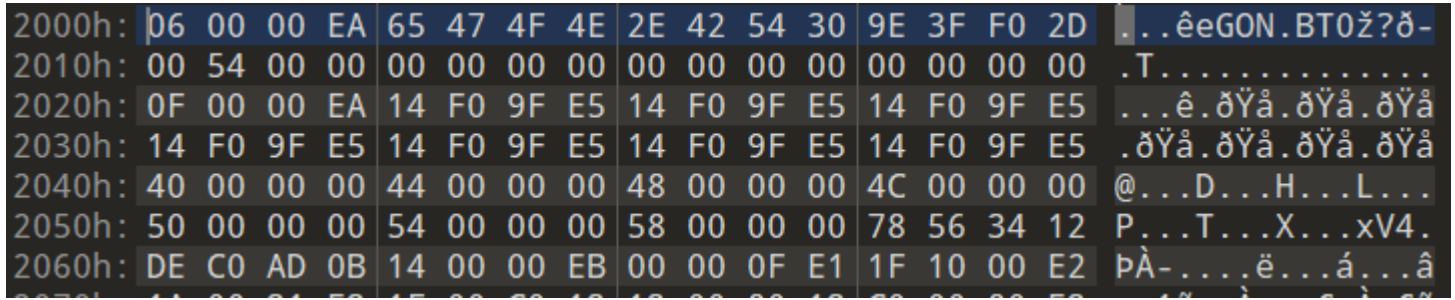
Allwinner芯片

- Sunxi 这里的x为数字，比如sun7i表示a20芯片，sun4i表示a10。
- BL1位于芯片内部，签名后的引导程序从0x20000开始载入，而不是像MBR一样从地址0开始。
- BL2（SPL）为签名（hash校验）后的uboot。

- 参考资料：https://linux-sunxi.org/A20
- 代码仓库：https://github.com/linux-sunxi

A20引导系统数据结构

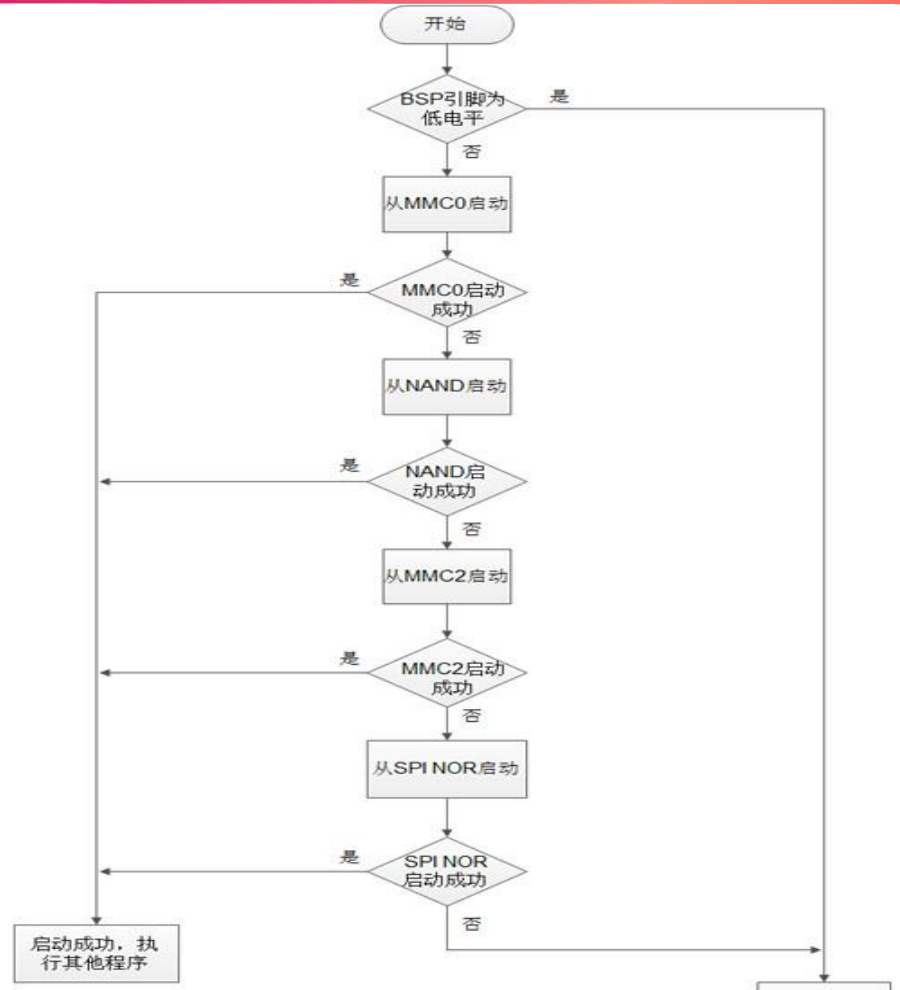
Offset	Name	Size	Notes
0x00	B_INS	4	Branch instruction to Code Starting Point
0x04	Magic	8	Ascii string "eGON.BT0" (No Null-terminated)
0x0c	Checksum	4	Simple 4-bytes **Checksum** (Before calculate checksum this must be 0x5F0A6C39)
0x10	Size	4	Size of Boot0, it's must be 8-KiB aligned in NAND and 512-Bytes aligned in MMC
0x14	Code	-	Code of SPL. The size depends on the processor and if it 's loaded from SPI, NAND or MMC



sudo dd if=u-boot-sunxi-with-spl.bin of=${card} bs=1024 seek=8

A20启动顺序

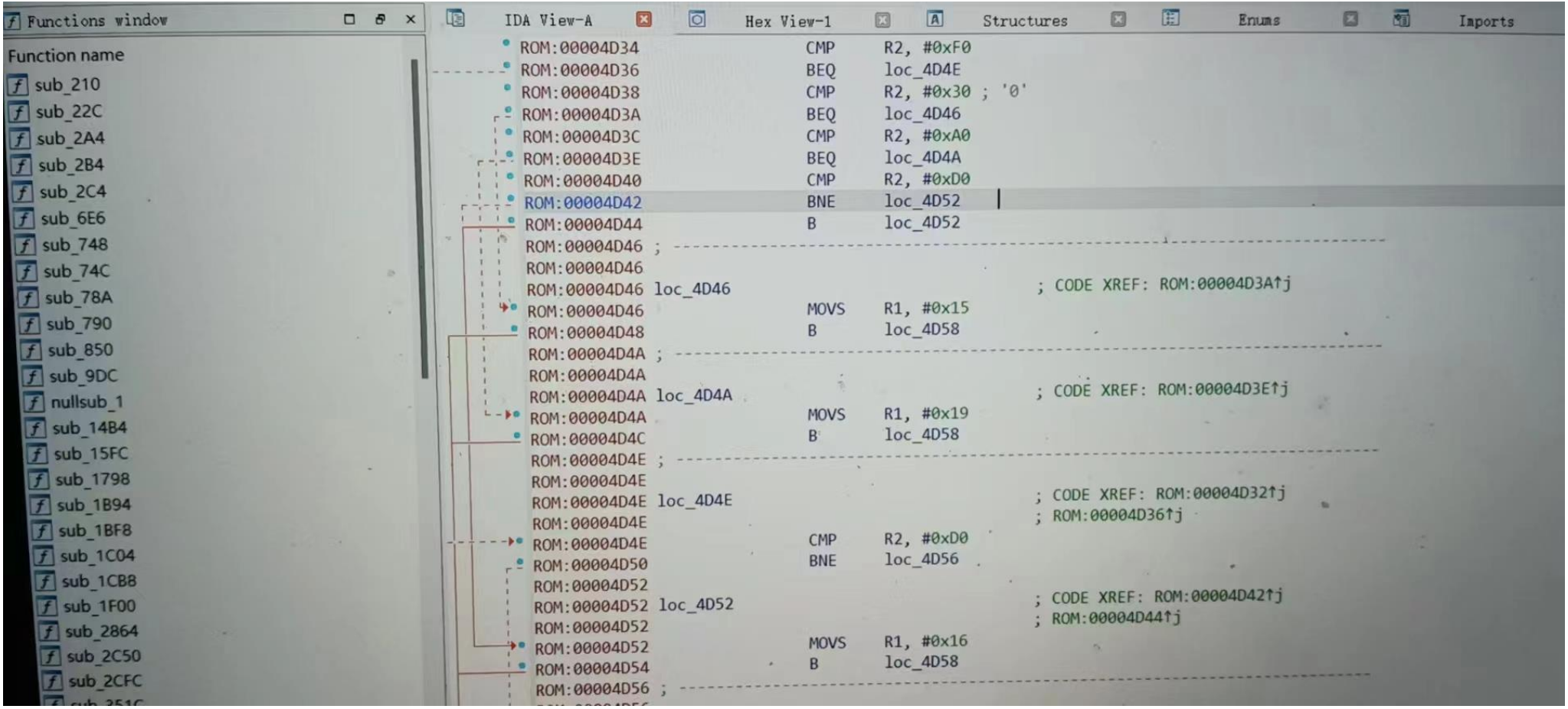
- A20支持从NAND Flash、SPI NOR Flash、SD card(SDC 0/2)和USB启动。当系统上电时，首先检测Boot Select Pin（BSP）管脚，如果为低电平，则直接从USB启动，否则尝试从MMC0启动，如果启动失败则尝试从NAND启动，如果启动失败则尝试从MMC2启动，如果启动失败则尝试从SPI NOR启动，如果启动失败则尝试从USB启动



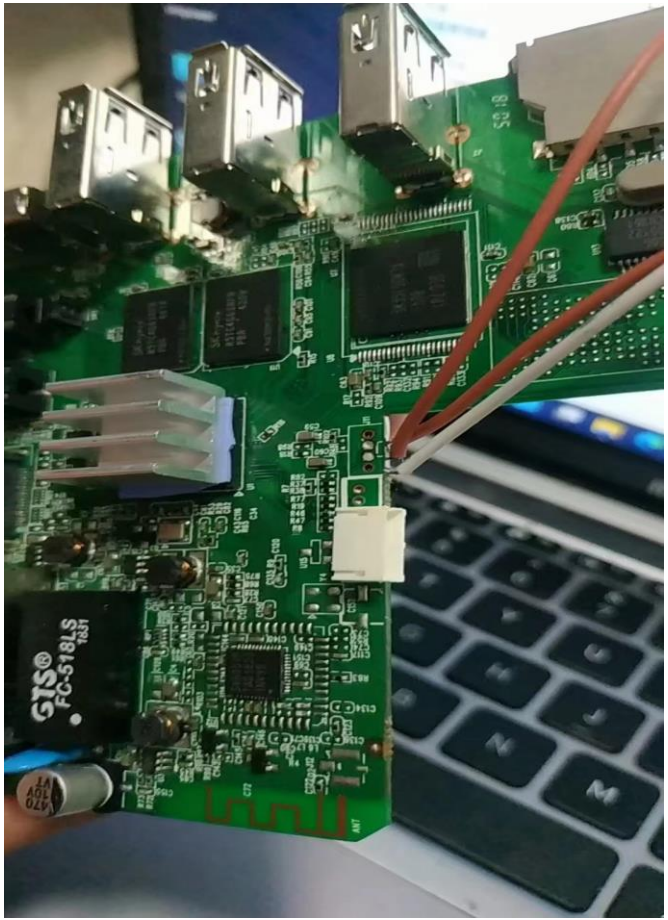
Amlogic芯片

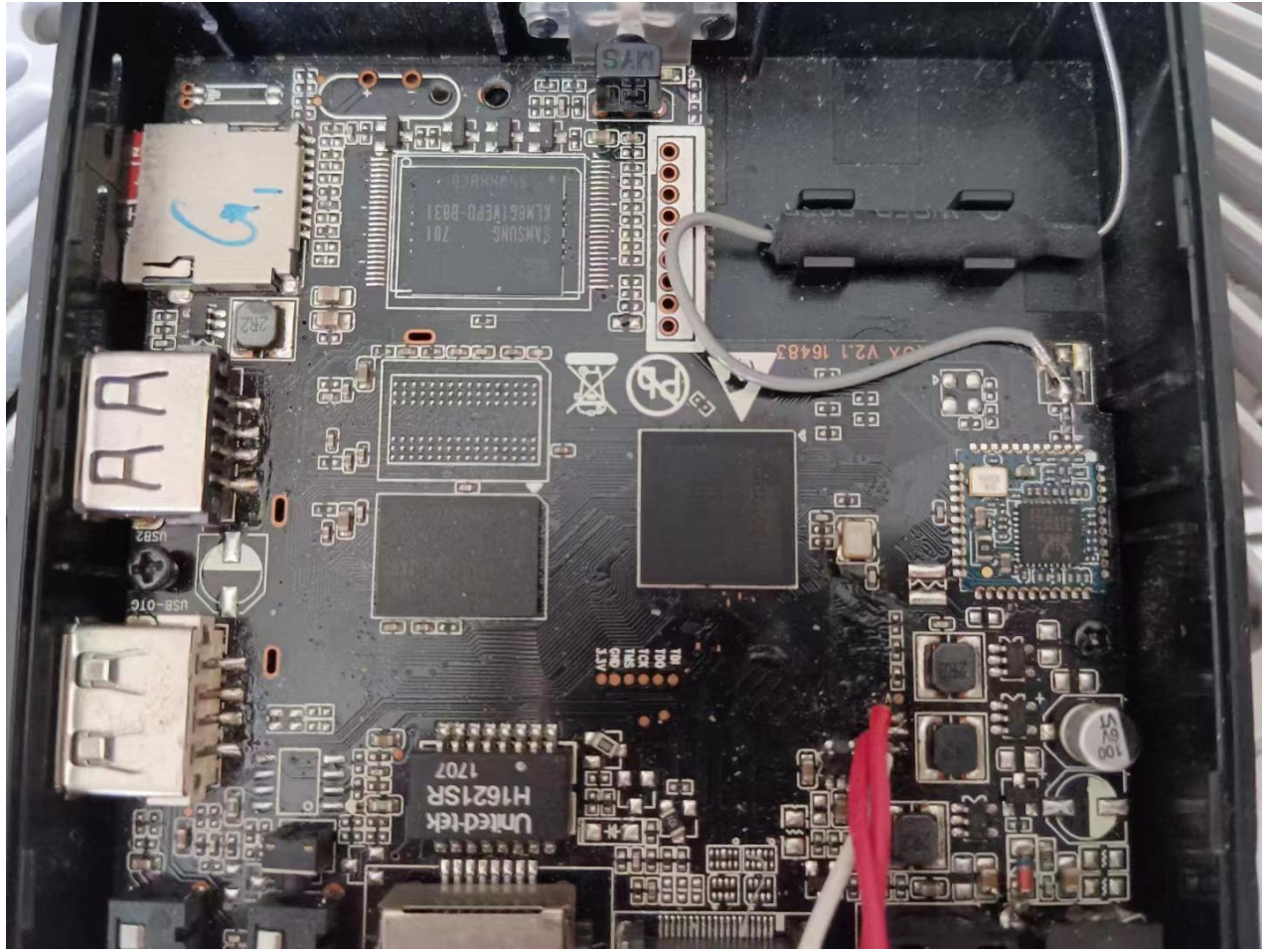
- bl1,bl2,bl31…
- 对uboot进行解耦，支持最新版本的uboot
- 对链接库的拆解
- 工具链
- bl1也在芯片内部，但是bl2闭源，bl3又分了3个阶段，只有bl33是uboot，这样的设计是做了解藕的，可以独立于uboot的源码，使得uboot可以跟随主线升级，但是一旦前面的任何一个环节被hack，整个bootloader就会沦陷，而hack的方式，实际上条件是要把前面的几个环节编译到bootloader中，想用热补丁的难度很大，因为在链接之前，它会把所有组件做签名，链接之后再做签名，bl1如果没有校验通过就不会加载bootloader，因此想在uboot之前介入，只有在链接前去hack，然后整个烧录到引导区。

通过IDA对各阶段的库进行逆向，可以植入代码到各个引导阶段，从而达到劫持引导区的目的。修改bl30跳转指令。



使得引导程序可以适配不同的电路板。



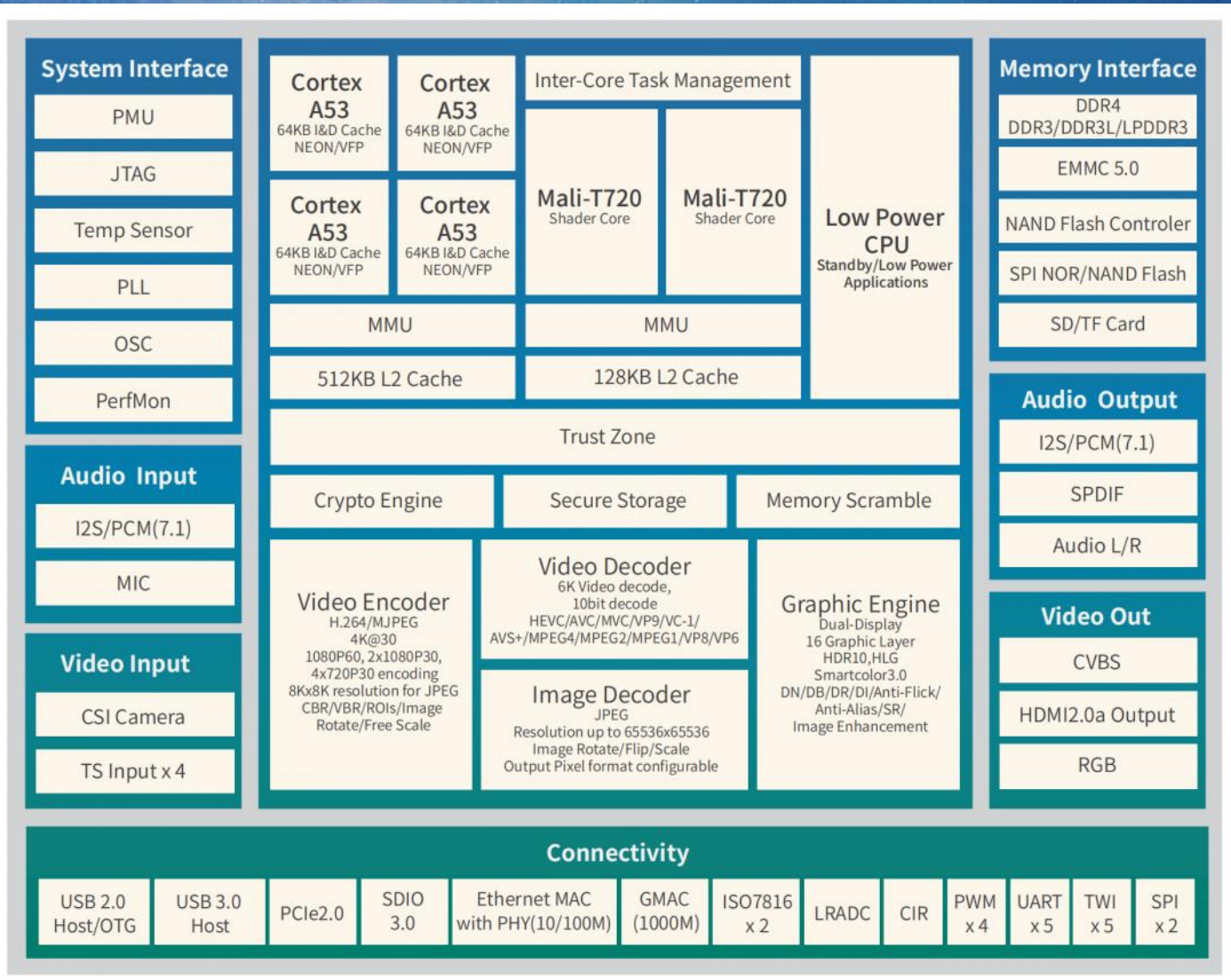


Uboot的环境变量

- UBOOT在启动时会打印出ENV的offset，经测试mmcblk0的偏移0x27400000处准确无误，在amlogic平台下，在uboot源码中也可以修改，通常如果为了hack，只做最小改动。
- UBOOT ENV存储数据结构。

- SOC芯片，指令没太大差别，真正的兼容性问题在于peripheral部分。总线，地址，寄存器等等差别大。
- 执行指令的模块直接拿来用的叫IP Core，通过HDL,verilog等描述，关联EDA软件。
- Hisi，rockchip，allwinner都用了新思科技的IP。
- 关于不同SOC在linux下的驱动，拿树莓派(博通)和PCDuino(全志)做了测试，后面举例。

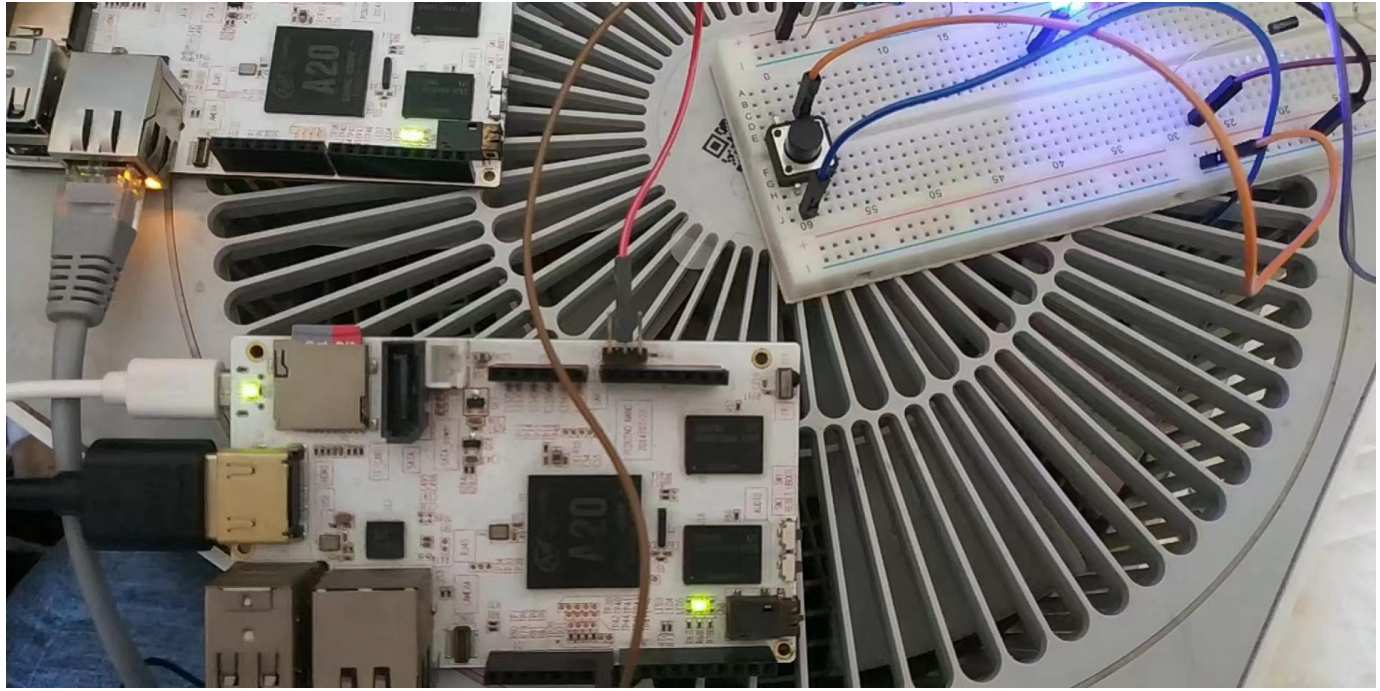
H6 SOC架构图


System Interface
PMU
JTAG
Temp Sensor
PLL
OSC
PerfMon
Audio Input
I2S/PCM(7.1)
MIC
Video Input
CSI Camera
TS Input x 4
Cortex A53
64KB I&D Cache
NEON/VFP
Cortex A53
64KB I&D Cache
NEON/VFP
Cortex A53
64KB I&D Cache
NEON/VFP
Cortex A53
64KB I&D Cache
NEON/VFP
Inter-Core Task Management
Mali-T720
Shader Core
Mali-T720
Shader Core
Low Power CPU
Standby/Low Power Applications
MMU
MMU
512KB L2 Cache
128KB L2 Cache
Trust Zone
Crypto Engine
Secure Storage
Memory Scramble
Video Encoder
H.264/MJPEG
4K@30
1080P60, 2x1080P30,
4x720P30 encoding
8Kx8K resolution for JPEG
CBR/VBR/ROIs/Image
Rotate/Free Scale
Video Decoder
6K Video decode,
10bit decode
HEVC/AVC/MVC/VP9/VC-1/
AVS+/MPEG4/MPEG2/MPEG1/VP8/VP6
Image Decoder
JPEG
Resolution up to 65536x65536
Image Rotate/Flip/Scale
Output Pixel format configurable
Graphic Engine
Dual-Display
16 Graphic Layer
HDR10,HLG
Smartcolor3.0
DN/DB/DR/DI/Anti-Flick/
Anti-Alias/SR/
Image Enhancement
Memory Interface
DDR4
DDR3/DDR3L/LPDDR3
EMMC 5.0
NAND Flash Controler
SPI NOR/NAND Flash
SD/TF Card
Audio Output
I2S/PCM(7.1)
SPDIF
Audio L/R
Video Out
CVBS
HDMI2.0a Output
RGB
Connectivity
USB 2.0 Host/OTG
USB 3.0 Host
PCIe2.0
SDIO 3.0
Ethernet MAC with PHY(10/100M)
GMAC (1000M)
ISO7816 x 2
LRADC
CIR
PWM x 4
UART x 5
TWI x 5
SPI x 2

举例：SOC的GPIO操作(官网下载手册)

- A20 pio: base addr 0x1c20800
- ph9 cfg: offset 0x100 bit 6:4　output: 0b001
- ph data: offset 0x10c bit 27:0

- 以上均为物理地址
- 可通过/dev/mem操作

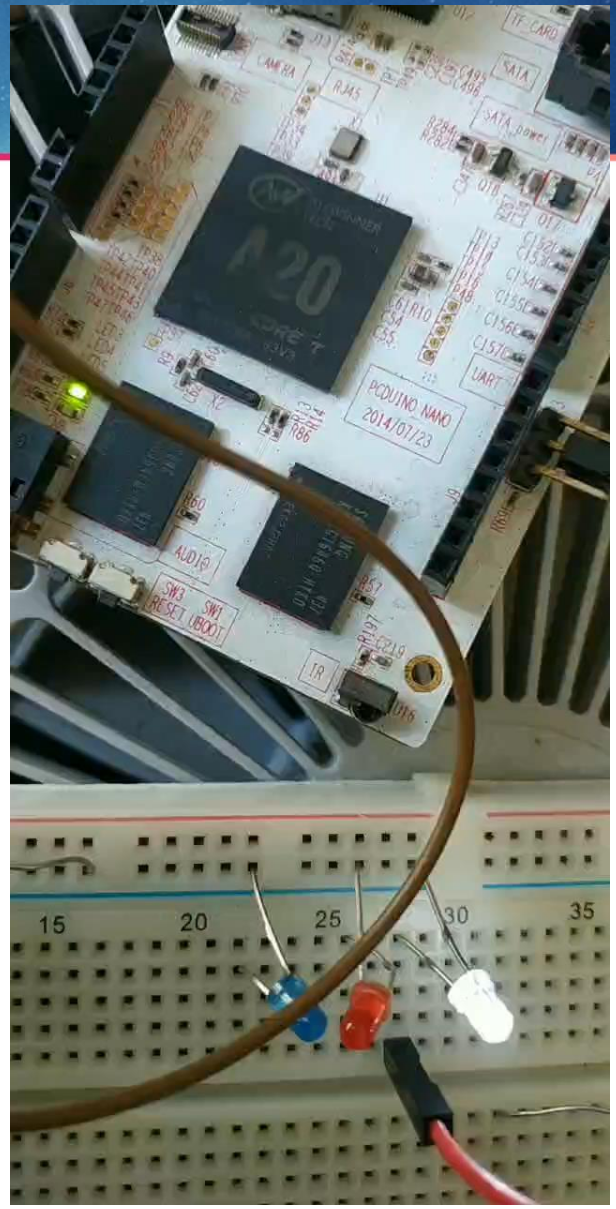


举例：A20的PWM操作

- A20 PB2: pwm0　board(PCDuinoNano3):J11-5
- Pio base addr: 1C20800
- offset 0x24　bit:10-8　af: 010
- Pwm base addr:0x1c20c00
- Ctrl register offset:0x200 0-9: 1(使能，chanel等等一大堆)
- period offset 0x204
- Entire cycle 31:16　*(Pulse-width modulation)频率*
- act 15:0 激活周期数

Linux驱动在板子上运行

- Pwm可以用于驱动无刷电机
- 进一步可以做飞控



03 对引导系统的攻击与防范

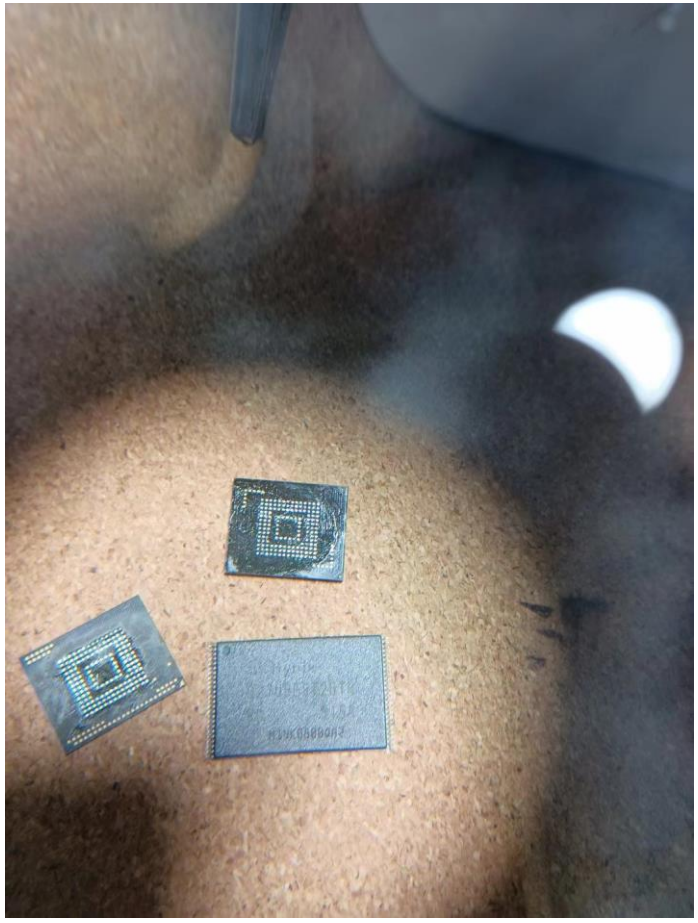
Bootkit，对引导系统进行攻击

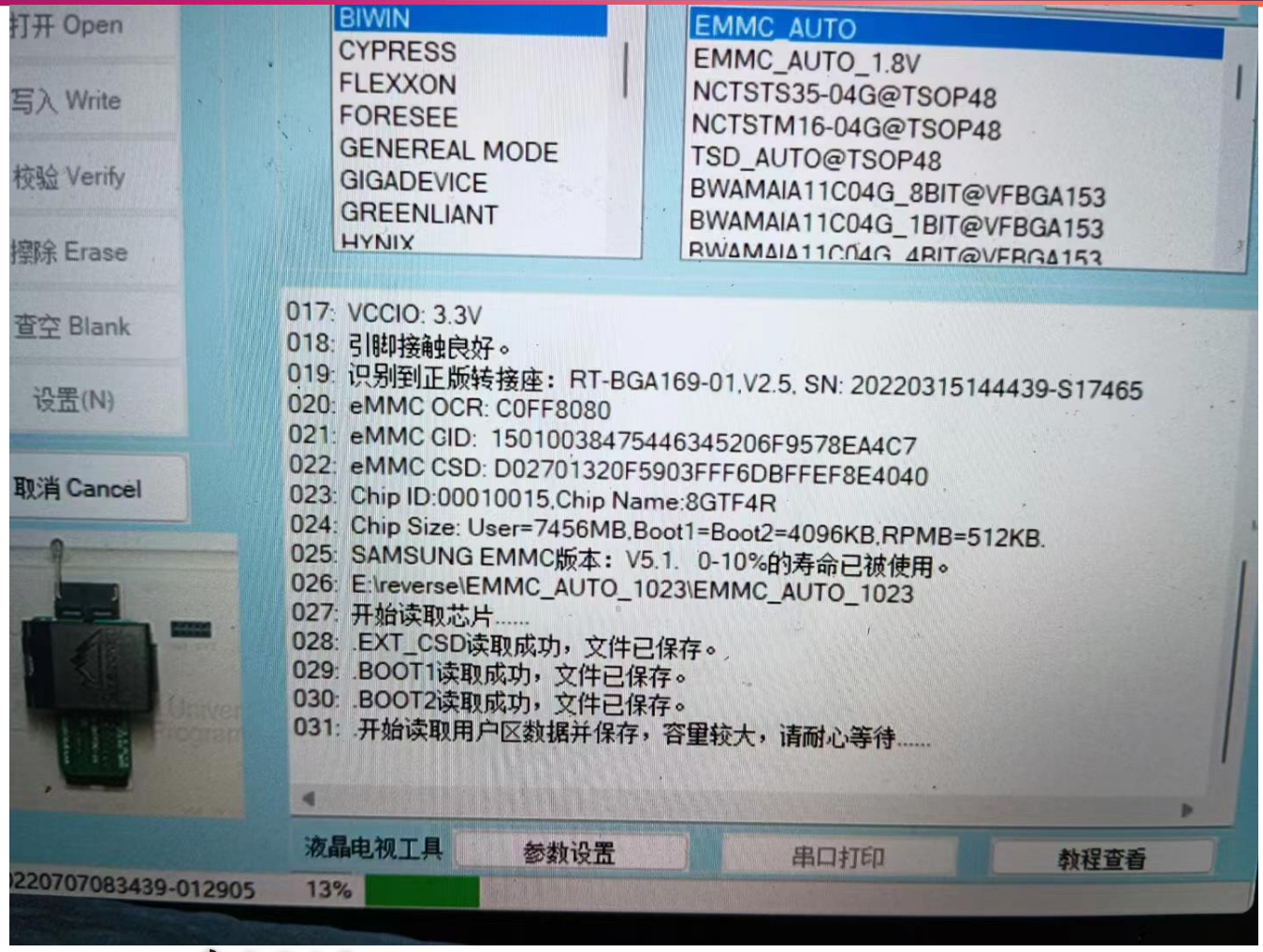
- MBR，唯一的验证就是扇区末尾的标记（55 AA），比如暗云3。
- UEFI，可劫持，用你的UEFI加载windows的引导模块，启动成功。
- UBOOT，Linux下取得root权限，就可以替换UBOOT。公网上可下载相关芯片的引导系统开发工具及文档。签名验签不是问题。

针对Uboot攻击

- TFTP劫持
- UART0命令行直接加载u盘程序
- 直接焊下EMMC，NandFlash存储芯片，根据手册进行读写，替换引导区
- 在linux/android操作系统中获取root权限，然后直接dd命令替换引导区
- 通过更改UBOOT环境变量，更改其启动行为，从而劫持正常引导
- 修改引导分区所存储的UBOOT启动脚本，更改其引导行为
- Android Recorery分区

拆存储芯片，直接读写固件





防范

- BL1在芯片内部，使用FPGA技术，让其可定制，或增加一个协处理器，签名验签算法或密钥自定义。
- UBOOT裁剪，去掉多余的功能，或自己写引导系统。
- 电路板不要暴露UART0针脚。
- 在操作系统内部对引导系统做校验。比如在分区之间的空位存放引导区的哈希，在操作系统运行期间对其校验。
- 填充物包裹电路板。

04 一些实践

善建者行，成其久远

- 让我们创造点什么吧！
- 安世盾个人防火墙！

通过研究引导系统而制作的新产品

- 支持多系统的硬件盒子(android,ubuntu,openwrt,emuelec…)
- 便携防火墙
- 路由器
- 多媒体中心
- 桌面系统
- 游戏机
- 未来将要加入：沙箱，漏扫等功能。

感谢大家的关注

安天冬训营 wtc.antiy.cn