



Tina Linux V85X

方案 FAQ

版本号: 1.0
发布日期: 2022.02.11

版本历史

版本号	日期	制/修订人	内容描述
1.0	2022.02.11	AWA0892	V85X 方案 FAQ



目 录

1 V85X 方案应用常见问题	1
1.1 概述	1
1.2 系统相关	1
1.2.1 V85x 是否已支持 RTOS 系统	1
1.2.2 异构系统中, RISC-V 如何与 A7 通信?	1
1.2.3 是否能独立于 Tina 构建系统打包	2
1.2.4 关于打包失败问题	2
1.2.5 V85x 方案原厂提供哪些算法	3
1.2.6 V853/V853S/V851/V851SE 芯片平台主要有哪些差异	3
1.2.7 如何获取物料支持列表	3
1.2.8 温控设置	4
1.2.9 如何查看 CPU 当前频率、如何修改 CPU 主频	5
1.2.9.1 查看 CPU 频率和策略	5
1.2.9.2 CPU 调频	5
1.2.10 如何在串口或者 ADB 登录时, 增加登录密码	6
1.2.10.1 ADB 权限功能	6
1.2.10.2 串口登录设置密码	6
1.2.11 是否有 package 包对应的 License 说明	7
1.2.12 支持休眠唤醒	7
1.2.13 产品要通过绑定 chipid 来授权, 需要获取唯一的 chipid	8
1.2.14 如何提高内核的打印等级	8
1.2.14.1 printk 的打印级别	8
1.2.14.2 查看内核启动初期的打印	8
1.3 DRAM	8
1.3.1 V85x 平台支持的 DDR3/DDR3L 的电压	8
1.3.2 如何导入 DRAM 物料	8
1.3.3 IPC 或 CDR 场景下内存评估	9
1.4 时钟相关	9
1.4.1 如何查看各模块时钟频率	9
1.4.2 如何修改 uart 的波特率	10
1.5 存储相关	13
1.5.1 为何 SPI Nand 存储可用空间比标称要小	13
1.5.2 存储切换方法	13
1.5.2.1 spinor 切换 spinand	13
1.5.2.2 spinand 切换 spinor	14
1.5.2.3 spinor 切换 emmc	15
1.5.2.4 spinor 切换 sdcard	16
1.5.3 是否支持 SPI Slave 模式	17
1.6 NPU 相关	17
1.6.1 NPU 问题: NPU 是否支持算子级别的接口	17
1.6.2 NPU 的资料如何获取, 是否有 License 问题	17

1.7 编码相关	17
1.7.1 编解码相关问题	17
1.8 Camera&&ISP 相关	18
1.8.1 双摄像头如何配置	18
1.8.2 ISP 设置区域曝光	18
1.9 显示相关	18
1.9.1 如何替换开机 Logo	18
1.9.2 如何确认 LCD 屏本身工作是否正常	18
1.10 GPIO 类（驱动能力设置）	18
1.10.1 芯片内部的 GPIO 是否带上下拉电阻	18
1.10.2 GPIO 的驱动能力如何配置	18
1.10.2.1 pinctrl 调试	18
1.10.2.2 Linux 标准 GPIO 调试	19
1.11 PMU 相关	20
1.11.1 PMU 调试命令	20
1.12 硬件信号测试相关	22
1.12.1 wifi 相关问题：XR819/XR829 调大发射功率	22
1.12.2 如何将 WIFI 的 MAC 地址固件下来（XR819/XR829）	23
1.12.3 EMI 相关问题：SDIO 展频、mipi-csi 展频、LCD 展频	24
1.12.3.1 DRAM 展频操作	24
1.12.3.2 SDIO 展频	24
1.12.3.3 mipi-csi 展频	25
1.12.3.4 mipi-dsi 展频 && CPU 接口的 LCD 展频	26
1.12.4 如何进入 USB 眼图测试	27
1.12.4.1 USB0 OTG Device 眼图测试:	27
1.12.4.2 USB0 OTG Host 眼图测试:	27
1.13 工具类	28
1.13.1 如何获取量产工具二次开发 SDK 包	28

插 图

1-1 E907-openamp	2
1-2 E907-openamp-client	2
1-3 V85x-feature	3
1-4 V85x-clk-summary	10
1-5 V85x-uart-boadrate	10
1-6 V85x-clk-tree	11
1-7 V85X-clk-tree-module	12
1-8 sunxi-pinctrl	19
1-9 xr829-0	22
1-10 xr829-1	23



1 V85X 方案应用常见问题

1.1 概述

本文主要记录在 V85x 等系列芯片在方案应用上，客户端常见的问题解答。适用于 V851、V853、V853s 和 V851se 等芯片及衍生平台。

1.2 系统相关

1.2.1 V85x 是否已支持 RTOS 系统

V85x 是双核异构系统，双核是指 Cortex-A7 + E907 RISC-V，其中 A7 运行的是 Linux 系统，E907 运行的是 melis 系统。暂无 A7 运行 RTOS 系统的计划。

1.2.2 异构系统中，RISC-V 如何与 A7 通信？

物理上，A7 与 E907 通过硬件 msgbox 进行通信，软件框架上，A7 上基于 Linux 标准的 rpmsg 驱动框架，E907 基于 openamp 异构通信框架。

A7 端：包含 remoteproc 驱动和 rpmsg 驱动；E907 端：包含 msgbox 通信和 openamp 通信；

其中 openamp 的框架如下：

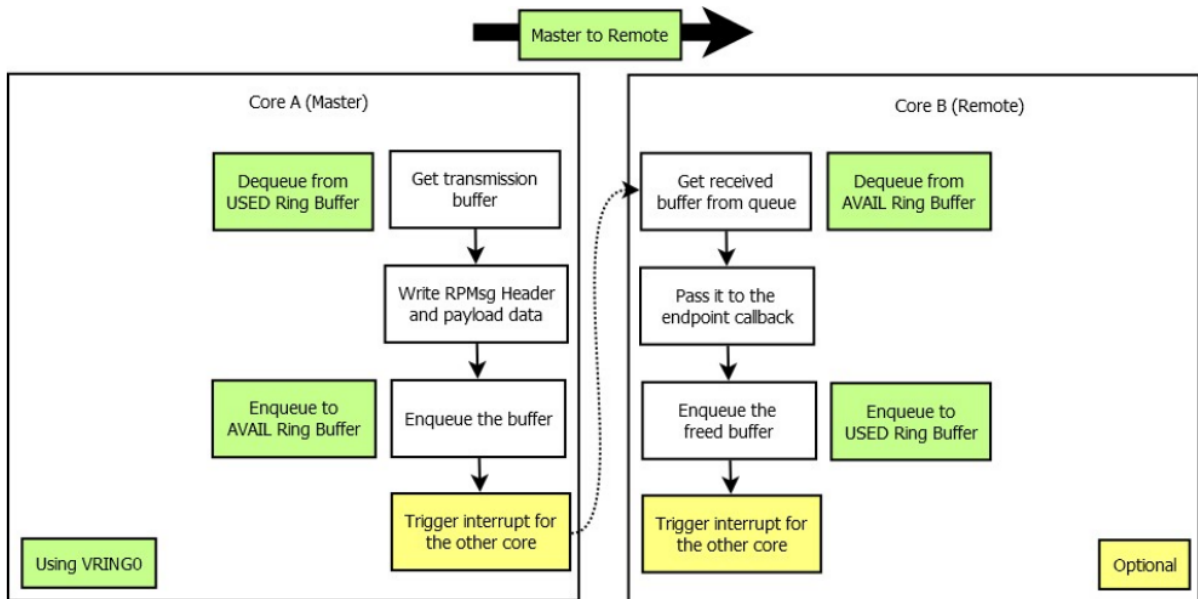


图 1-1: E907-openamp

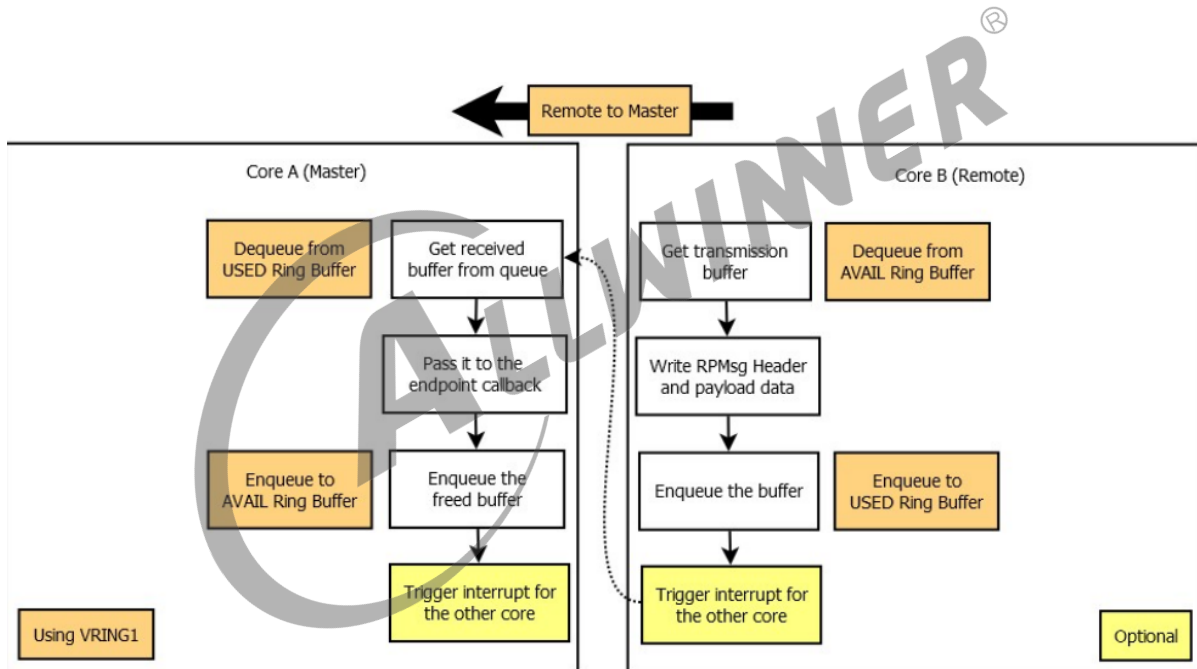


图 1-2: E907-openamp-client

1.2.3 是否能独立于 Tina 构建系统打包

可独立打包，可通过 Aservice 提交需求单独提供参考范例。

1.2.4 关于打包失败问题

pack 打包时出现ERROR: merge_full_img failed

修改device/config/chips/v853/configs/perf1/linux/sys_partition.fex或者sys_partition_nor.fex对应分区的大小。

1.2.5 V85x 方案原厂提供哪些算法

音频类：

- 支持 EQ/DRC 算法、音效组件及 PC 端工具
- 单 MIC 降噪和回声消除算法

图像类：

- 移动侦测算法、人形检测算法、区域入侵检测算法

1.2.6 V853/V853S/V851/V851SE 芯片平台主要有哪些差异

		智慧视觉			
芯片型号		V851	V851SE	V853	V853S
分档		4M插值5M	4M插值5M	5M(3072x1721)	5M(3072x1721)
AI	算力	0.5T	0.5T	1T	1T
封装	封装尺寸	9x9 QFN88-0.35mm	9x9 QFN88-0.35mm	12X12 BGA318-0.65mm	12X12 BGA318-0.65mm
CPU	core	A7-900Mhz E906-600MHz	A7-900Mhz E906-600MHz	A7-1.2GHz E907-600MHz	A7-1.2GHz E907-600MHz
DDR	类型	SIP-64MB-DDR2 (1.5V)	SIP-64MB-DDR2 (1.5V)	外挂DDR	SIP 128MB DDR3/DDR3L
VE	H264编码能力	5M @20fps/4M @30fps	5M @20fps/4M @30fps	5M @25fps/4M @30fps	5M @25fps/4M @30fps
	H265编码能力	5M @20fps/4M @30fps	5M @20fps/4M @30fps	5M @25fps/4M @30fps	5M @25fps/4M @30fps
ISP	性能	4M	4M	5M	5M
Video out	接口类型	1x8bit LCD输出	1x8bit LCD输出	1x18bit LCD输出	1x18bit LCD输出
		1xBT656输出	1xBT656输出	1xBT1120输出	1xBT1120输出
		1xMIPI-DSI 2lane	1xMIPI-DSI 2lane	1xMIPI-DSI 4lane	1xMIPI-DSI 4lane
Video in	接口类型	1xMIPI-CSI 4lane 可拆分2x2lane 1x10/并口CSI（复用）	1xMIPI-CSI 4lane 可拆分2x2lane 1x10/并口CSI（复用）	1xMIPI-CSI 4lane 可拆分2x2lane 1x8/10/12并口CSI 1xBT1120输入	1xMIPI-CSI 4lane 可拆分2x2lane 1x8/10/12并口CSI 1xBT1120输入
		分辨率	4M @30FPS	4M @30FPS	5M @30FPS
	网口	RMII	SIP EPHY	RMII接口(或SIP EPHY)	RMII接口(或SIP EPHY)

图 1-3: V85x-feature

1.2.7 如何获取物料支持列表

一号通->文档中心->V85x/V85x-IPC/Hardware硬件类文档/硬件物料清单/

1.2.8 温控设置

V85x 芯片的 ARM、NPU、VE、E907 RISC-V 有温控传感器，可以实时监控这几个核的温度，并通过算法来控制 cpu 频率从而控制 cpu 的温度。每个产品的硬件设计和模具不同对应的散

热情况也不同，可以通过 dts 中的如下配置进行适当的调整温控参数来适配产品：
设置温控开启的温度：

- dts 的文件为:lichee/linux-4.9/arch/arm/boot/dts/sun8iw21p1.dtsi

```
thermal-zones {
    cpu_thermal_zone {
        polling-delay-passive = <500>;
        polling-delay = <1000>;
        thermal-sensors = <&ths 2>;
        sustainable-power = <68>;

        cpu_trips: trips {
            cpu_threshold: trip-point@0 {
                temperature = <70000>;
                type = "passive";
                hysteresis = <0>;
            };
            cpu_target: trip-point@1 {
                temperature = <90000>;
                type = "passive";
                hysteresis = <0>;
            };
            cpu_crit: cpu_crit@0 {
                temperature = <110000>;
                type = "critical";
                hysteresis = <0>;
            };
        };

        cooling-maps {
            map0 {
                trip = <&cpu_target>;
                cooling-device = <&cpu0
                    THERMAL_NO_LIMIT
                    THERMAL_NO_LIMIT>;
                contribution = <1024>;
            };
        };
    };

    npu_thermal_zone {
        polling-delay-passive = <0>;
        polling-delay = <0>;
        thermal-sensors = <&ths 0>;
    };

    ve_thermal_zone {
        polling-delay-passive = <0>;
        polling-delay = <0>;
        thermal-sensors = <&ths 1>;
    };
};
```

```
};  
};
```

- 可以通过此节点查看各个核的 sensor 温度

```
root@TinaLinux:/sys/class/thermal# cat /sys/class/thermal/*/type  
thermal-cpufreq-0  
cpu_thermal_zone  
npv_thermal_zone  
ve_thermal_zone  
battery  
root@TinaLinux:/sys/class/thermal# cat /sys/class/thermal/*/temp  
30380          /* 温度单位为mc, 就是30.380度*/  
30520  
30590  
0  
root@TinaLinux:/sys/class/thermal#
```

1.2.9 如何查看 CPU 当前频率、如何修改 CPU 主频

1.2.9.1 查看 CPU 频率和策略

```
查看cpu当前频率: cat /sys/devices/system/cpu/cpu0/cpufreq/cpuinfo_cur_freq
```

```
查看cpu调频策略: cat /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor  
performance
```

表示当前调频策略为 performance, 它会以当前最大频率运行。其他常用的调频策略, 如 interactive, 会根据实际负载调节频率。一般常用的就是这两种, 如果了解其他调频策略, 可以查看内核源码中 Documentation/cpu-freq/governors.txt 该文档, 或者网上资料。

1.2.9.2 CPU 调频

```
#设置用户调频模式  
echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor  
  
#查看可调频率  
cd /sys/devices/system/cpu/cpu0/cpufreq/  
cat scaling_available_frequencies  
  
eg: 720000 816000 912000 960000 1008000 1200000 1320000 1440000 1512000  
  
#设置CPU频率:  
echo 720000 > scaling_setspeed      #设置CPU主频为720M  
echo 1200000 > scaling_setspeed     #设置CPU主频为1200M
```

1.2.10 如何在串口或者 ADB 登录时，增加登录密码

1.2.10.1 ADB 权限功能

ADB 的用法可参考一号通->文档中心->V85X->V85X IPC->Software软件类文档->基础组件开发指南->《Tina_Linux_USB_开发指南.pdf》

该功能主要用于权限管理，只有经过公私钥认证后才能进行 adb 通信。

1.2.10.1.1 配置 ADB 使用前需要 make menuconfig 选中对应功能：

```
Allwinner --->
<*> adb..... adb for Tina Linux
<*> abdb_auth_service..... adb auth service for Tina Linux
```

默认开机就会使能该功能，执行 adb 命令时会进行公私钥校验。

1.2.10.1.2 公钥的安装：

默认允许安装所有公钥，可以通过修改tina/package/utils/adb/auth/aw_adb_auth_service.c
取消下面注释
/*aw_adbd_install_pubkey_detector(publickey_detector); */
然后在publickey_detector函数中判断是否安装公钥，返回false则不安装
公钥的安装路径：/mnt/UDISK/adb_keys

如果/mnt/UDISK/adb_keys 下面没有对应的公钥，并且不允许安装新公钥，这时候 adb 功能就不能使用。

1.2.10.1.3 注意事项： 1.Windows 下面公私钥的路径：

```
%USERPROFILE%\android\adbkey // 私钥
%USERPROFILE%\android\adbkey.pub // 公钥
```

2.Ubuntu 下面公私钥的路径：

```
用户目录/.android/adbkey // 私钥
用户目录/.android/adbkey.pub // 公钥
```

1.2.10.2 串口登录设置密码

1.2.10.2.1 配置 busybox

```
make menuconfig选择

Base system ->
<*> busybox .....
[*] Customize busybox optionLogin/Password Management Utilities ->
[*] login (NEW)
```

1.2.10.2.2 修改启动脚本 package/base-files/files/etc/inittab

```
vim /etc/inittab
::askconsole:/bin/ash -login
改为
::askconsole:/bin/login
```

1.2.10.2.3 或修改 vi target/allwinner/v853-xxx/xx/etc/inittab

```
ttyMSM0::askfirst:/bin/login
```

inittab 一般在方案目录下都会有，例如 v853-perfl 方案：

```
target/allwinner/v853-perfl/busybox-init-base-files/etc/inittab
```

1.2.10.2.4 root 密码默认设置

```
package/base-files/files/etc/shadow
默认密码配置为tina, 对应/etc/shadow
root:91rMiZzGliXHm:1:0:99999:7:::
```

1.2.11 是否有 package 包对应的 License 说明

Allwinner 有整理 package 包对应的 License 声明，如需要可通过 Aservice 索取。

1.2.12 支持休眠唤醒

可支持 super standby 和 Normal standby。

super standby 唤醒源支持：RTC、USB 插入/拔出、Gesensor、按键。

Normal Standby 唤醒源支持：Wifi ping 唤醒、RTC、USB 插入/拔出、Gesensor、按键、GPIO。

1.2.13 产品要通过绑定 chipid 来授权，需要获取唯一的 chipid

cat /sys/class/sunxi_info/sys_info 读取 sunxi_serial, sunxi_serial 具有唯一性。

1.2.14 如何提高内核的打印等级

可通过读写 /proc/sys/kernel/printk 文件，读取和修改控制台的日志级别

```
echo 8 > /proc/sys/kernel/printk
```

或修改 env.cfg 将 loglevel 置成 8。

1.2.14.1 printk 的打印级别

```
#define KERN_EMERG      "<0>" /* system is unusable */
#define KERN_ALERT      "<1>" /* action must be taken immediately */
#define KERN_CRIT       "<2>" /* critical conditions */
#define KERN_ERR         "<3>" /* error conditions */
#define KERN_WARNING     "<4>" /* warning conditions */
#define KERN_NOTICE      "<5>" /* normal but significant condition */
#define KERN_INFO        "<6>" /* informational */
#define KERN_DEBUG       "<7>" /* debug-level messages */
```

1.2.14.2 查看内核启动初期的打印

```
make kernel_menuconfig
Kernel hacking ---> Kernel low-level debugging functions -->Early printk
```

1.3 DRAM

1.3.1 V85x 平台支持的 DDR3/DDR3L 的电压

DDR3 是 1.5V 的，DDR3L 是 1.35V，LPDDR3 是 1.2V，不过 V853 只支持 16bit DDR，LPDDR3 是 32bit，不支持 LPDDR3。

1.3.2 如何导入 DRAM 物料

需求导入：通过 Aservice 系统提交物料调试需求

- 上传 DRAM 相关的 SPEC
- 邮寄相关物料及平台 (4pcs + 4pcs board)

验证：稳定性和可靠性测试原厂提供更新的 dram 驱动后，客户需要按照 一号通->文档中心->V85X->V85X IPC->Software软件类文档->基础组件开发指南->《全志R&V系列Tina DRAM物料兼容性测试操作指南V1.0》进行可靠性和稳定性验证，通过后反馈给原厂加入支持列表中。

1.3.3 IPC 或 CDR 场景下内存评估

有参考版本，可通过 Aservice 提交需求单独提供。

1.4 时钟相关

1.4.1 如何查看各模块时钟频率

```
mount -t debugfs none /sys/kernel/debug  
cat /sys/kernel/debug/clk/clk_summary
```

如示例：

```

root@TinaLinux:/# cat /sys/kernel/debug/clk/clk_summary

```

clock	enable_cnt	prepare_cnt	rate	accuracy	phase
osc48m	0	0	48000000	0 0	
osc48md4	0	0	12000000	0 0	
usbhci0_12m	0	0	12000000	0 0	
hoscdv32k	0	0	32768	0 0	
hosc32k	0	0	32768	0 0	
hosc	8	9	24000000	0 0	
sdmmc1_mod	0	0	200000	0 0	
csi_master0	0	0	24000000	0 0	
sdmmc0_mod	0	0	800000	0 0	
cpurapbs0	1	1	24000000	0 0	
cpurcpus	0	0	24000000	0 0	
cpurppu	1	1	24000000	0 0	
cpurtwd	0	0	24000000	0 0	
cpurahbs	0	0	24000000	0 0	
rtc_spi	0	0	1200000	0 0	
cpurrtc	0	0	24000000	0 0	
fanout_24m	0	0	24000000	0 0	
e907	0	0	24000000	0 0	
e907_axi	0	0	12000000	0 0	
isp	0	0	24000000	0 0	
csi_master2	0	0	24000000	0 0	
csi_master1	0	0	24000000	0 0	
usbphy0	1	1	24000000	0 0	
gpadc	1	1	24000000	0 0	
spif	0	0	24000000	0 0	
spi2	0	0	24000000	0 0	
spi1	0	0	24000000	0 0	
sdmmc2_rst	0	0	24000000	0 0	
sdmmc2_bus	0	0	24000000	0 0	
sdmmc2_mod	0	0	24000000	0 0	
sdmmc1_rst	0	0	24000000	0 0	
sdmmc1_bus	0	0	24000000	0 0	
sdmmc0_rst	0	0	24000000	0 0	
sdmmc0_bus	0	0	24000000	0 0	
dbgsys	0	0	24000000	0 0	
avs	0	0	24000000	0 0	
apb1	3	3	24000000	0 0	
twi4	1	1	24000000	0 0	
twi3	0	0	24000000	0 0	
twi2	0	0	24000000	0 0	
twi1	0	0	24000000	0 0	

图 1-4: V85x-clk-summary

1.4.2 如何修改 uart 的波特率

V85x 串口默认波特率为 115200，某些场景下需要使用其它波特率进行数据传输，如蓝牙播放音乐的场景通常需要将波特率设置为 1500000。uart 内核驱动 (lichee/linux-4.9/drivers/tty/serial/sunxi-uart.c) 提供了以下的提示信息：

```

/*
 * uart baudrate and apb2 clock config selection
 * We should select an apb2 clock as low as possible
 * for lower power consumption, which can satisfy the
 * different baudrates of different ttys applications.
 *
 * the reference table as follows:
 * pll6 600M
 * apb2div
 * apbclk
 * 0 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5
 * 24000000 30000000 31578947 33333333 35294117 37500000 40000000 42857142 46153846 50000000 54545454 60000000 66666666 75000000 85714285 100000000 120000000
 * 115200 * * * * * * * * * * * * * * * *
 * 230400 * * * * * * * * * * * * * * * *
 * 380400 * * * * * * * * * * * * * * * *
 * 460800 * * * * * * * * * * * * * * * *
 * 921600 * * * * * * * * * * * * * * * *
 * 1000000 * * * * * * * * * * * * * * * *
 * 1500000 * * * * * * * * * * * * * * * *
 * 1750000 * * * * * * * * * * * * * * * *
 * 2000000 * * * * * * * * * * * * * * * *
 * 2500000 * * * * * * * * * * * * * * * *
 * 3000000 * * * * * * * * * * * * * * * *
 * 3250000 * * * * * * * * * * * * * * * *
 * 3500000 * * * * * * * * * * * * * * * *
 * 4000000 * * * * * * * * * * * * * * * *
 */

```

图 1-5: V85x-uart-boadrate

从上述注释，可了解到 uart 波特率与时钟配置的关系，apbclk 横轴为 apb 各个频率，纵轴为该

时钟支持的波特率。默认时钟为 24M，由此可见，在不修改时钟配置的前提下，我们可使用最大的波特率为 1.5M。

- 修改时 uart 钟源

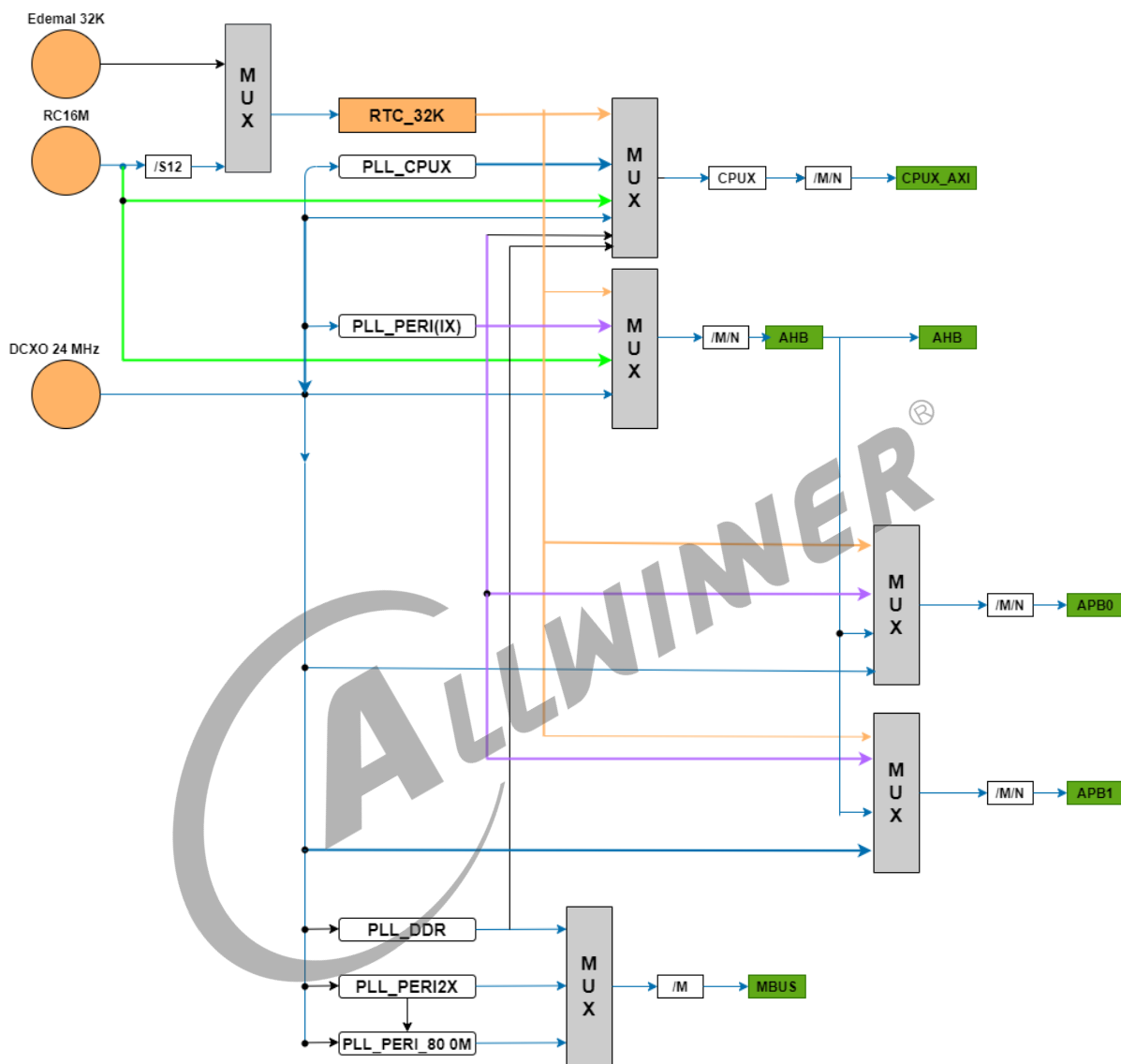


图 1-6: V85x-clk-tree

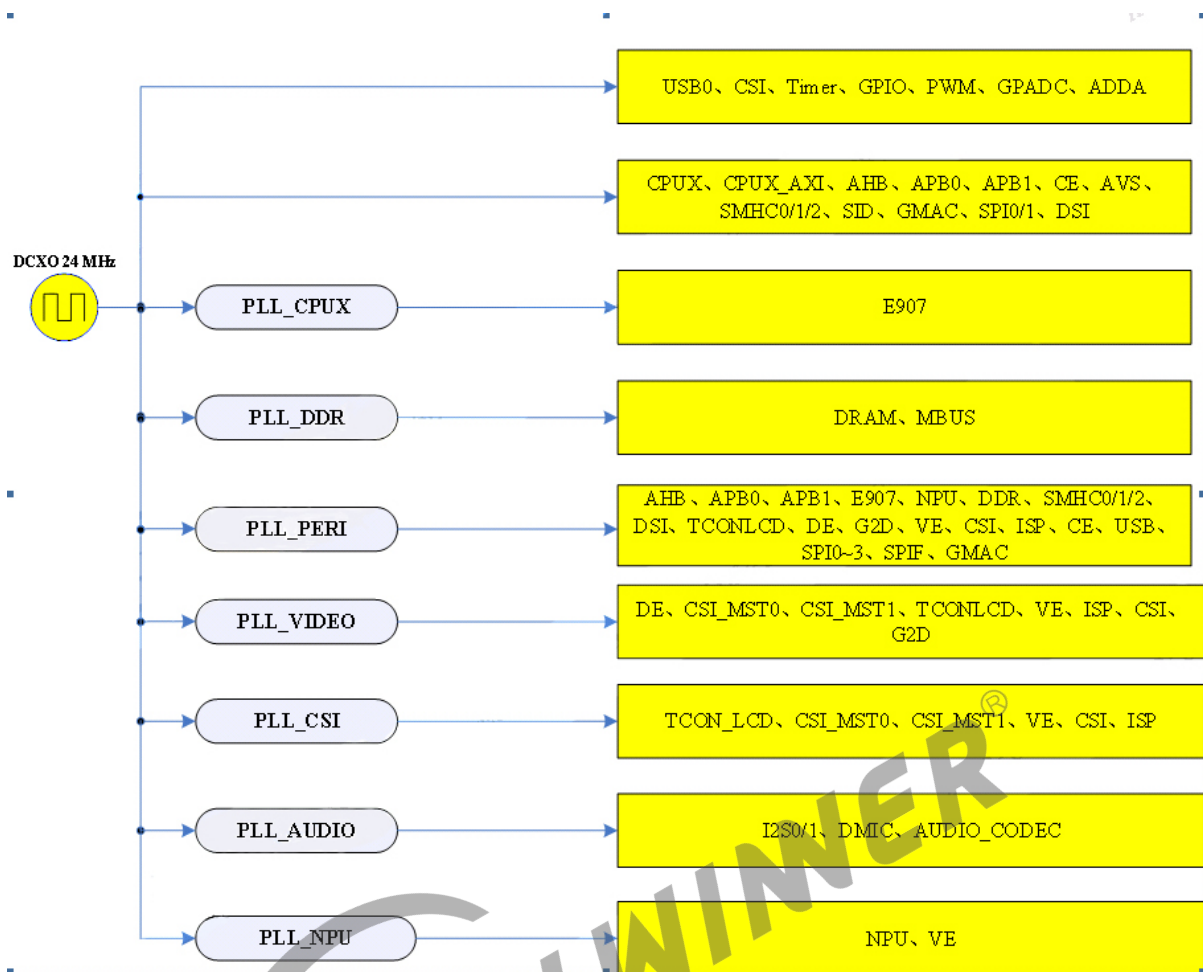


图 1-7: V85X-clk-tree-module

上图为时钟源和时钟树，根据目标时钟修改 DTS 中 UART 的时钟源，其中需要注意：

Clocks 需要按照芯片 spec 配置 (或者查看 clk 代码)，第一个参数为 uart clk, 第二个为 apb2 clk, 第三个为 apb2 的时钟源 pll_periph0。这里可配置的时钟源不同平台有可能有差异

Clock-frequency 并不是任意频率均可以设置，它是由时钟源 pll_periph0 分频得到，其中分频系数需要查看 spec, 例如 sun8iw15 的，factor N:1,2,4,8, 而 factor M:1~4, 因此 50M 可以由 N=4,M=3 获取得到： $600/4/3=50$

如果设置的频率无法准确分频得到，则会设置附近的一个值，例如 30M，实际分频： $N=4,M=4, 600/4/4=37.5M$

这种修改方法，是在内核初始化 uart 的时候才生效，如果想提前，得改动 uboot, 或者 boot0 的代码 (主要就是设置 apb2 的时钟源，及分频系数)

1.5 存储相关

1.5.1 为何 SPI Nand 存储可用空间比标称要小

如 Nand flash 总大小是 128M Bytes, Nand 驱动本身会保留 1/8 到 1/10 左右的空间, 包括用于 Nand 管理、预留给 Boot0/uboot 的空间, 如果 nand 存在坏块, 空间也将相应的减小;

1.5.2 存储切换方法

SDK 切换存储介质需要修改 board.dts、sys_config.fex、内核配置、TINA 系统配置。另外, 在 spinor 存储介质下, 通过 u-boot-sun8iw21p1.bin 进行烧录, u-boot-spinor-sun8iw21p1.bin 启动, 使用 sys_partition_nor.fex 作为分区表。在非 spinor 介质 (spinand、emmc、sdnand), 通过 u-boot-sun8iw21p1.bin 进行烧录和启动, 使用 sys_partition.fex 作为分区表。SUN8IW21 的各份 SDK 默认使用 spinor 作为存储介质, 下文将介绍 spinor 切换 spinand、spinand 切换 spinor、spinor 切换 emmc、spinor 切换 sdnand 四种切换方式。

- sys_config.fex 中只要关注 storage_type 选项, 这个选项决定使用哪一种存储介质, 配置后在 pack 过程, 会自动打包对应介质的二进制文件。

[target]

storage_type = xxx 其中 storage_type | 0:nand | 1:sd | 2:emmc | 3:spinor | 4:emmc3 | 5:spinand | 6:sd1 |

1.5.2.1 spinor 切换 spinand

1.5.2.1.1 sys_config.fex 修改

```
[target]
- storage_type = 3
+ storage_type = 5
```

1.5.2.1.2 board.dts/uboot-board.dts 修改

```
&spi0 {
-   status = "disabled";
+   status = "okay";
    spi-nand@0 {
-       status="disabled";
+       status="okay";
    };
};
```

1.5.2.1.3 uboot 编译

```
编译u-boot-sun8iw2lp1.bin, spinand使用u-boot-sun8iw2lp1.bin进行烧录、启动。
cboot切换到uboot目录, 编辑configs/sun8iw2lp1_defconfig
CONFIG_SUNXI_UBIFS=y    //SDK发布默认已选上
执行mboot              //编译
确保device/config/chips/vxxx/bin/u-boot-sun8iw2lp1.bin已更新
```

1.5.2.1.4 内核配置

```
make kernel_menuconfig 层层选中
内核驱动配置
Device Driver --->
  Memory Technology Device (MTD) support --->
    sunxi-nand --->
      <*> AWINAND CHOICE (Allwinner MTD SPINAND Device Support) --->
        *- Enable UBI - Unsorted block images --->
          [*] Read-only block devices on top of UBI volumes

内核文件系统配置
File systems --->
  [*] Miscellaneous filesystems --->
    <*> UBIFS file system support
```

1.5.2.1.5 TINA 环境配置

```
make menuconfig 层层选中
Target Images --->
  Boot (SD Card) Kernel format (boot.img) ---> //默认选中
  [ ] For storage less than 32M, enable this when using ota //取消勾选

Global build settings --->
  [*] Strip unnecessary functions from libraries //取消勾选

Utilities --->
  <*> mtd-utils --->
    <*> mtd-utils-mkfs.ubifs
```

1.5.2.2 spinand 切换 spinor

1.5.2.2.1 sys_config.fex 修改

```
[target]
- storage_type = 5
+ storage_type = 3
```

1.5.2.2.2 board.dts/uboot-board.dts 修改

```
&spi0 {
-   status = "okay";
+   status = "disabled";
  spi-nand@0 {
```

```
-     status="okay";  
+     status="disabled";  
    };  
};
```

1.5.2.2.3 内核配置

```
make kernel_menuconfig 层层选中  
内核驱动配置  
Device Driver --->  
    Memory Technology Device (MTD) support --->  
        sunxi-nand ---> //取消勾选  
    Self-contained MTD device drivers --->  
        <*> Support most SPI Flash chips (AT26DF, M25P, W25X, ...) //确认已选中  
        <*> SPI-NOR device support --->  
内核文件系统配置  
File systems --->  
    [*] Miscellaneous filesystems --->  
        <*> Journalling Flash File System v2 (JFFS2) support
```

1.5.2.2.4 TINA 环境配置

```
make menuconfig 层层选中  
Target Images --->  
    [*] For storage less than 32M, enable this when using ota //选中  
Global build settings --->  
    [*] Strip unnecessary functions from libraries //选中  
Utilities --->  
    <*> mtd-utils  
        <*> mtd-utils-mkfs.jffs2 //选中
```

1.5.2.3 spinor 切换 emmc

1.5.2.3.1 sys_config.fex 修改

```
[target]  
- storage_type = 3  
+ storage_type = 2
```

1.5.2.3.2 内核配置

```
make kernel_menuconfig 层层选中  
[*] Enable the block layer --->  
    [*] Support for large (2TB+) block devices and files  
    [*] Block layer SG support v4  
Device Drivers --->  
    < > Memory Technology Device (MTD) support ---> //取消勾选  
    [*] Block devices ---> //确认勾选  
File systems --->  
    <*> The Extended 4 (ext4) filesystem
```

1.5.2.3.3 TINA 环境配置

```
make menuconfig 层层选中
Target Images --->
  Boot (SD Card) Kernel format (boot.img) ---> //默认选中
  [ ] For storage less than 32M, enable this when using ota //取消勾选

Global build settings --->
  [ ] Strip unnecessary functions from libraries //取消勾选

Utilities --->
  Filesystem --->
    <*> e2fsprogs //选中
```

1.5.2.4 spinor 切换 sdnand

1.5.2.4.1 sys_config.fex 修改

```
[target]
- storage_type = 3
+ storage_type = 1
```

1.5.2.4.2 board.dts 修改

```
&sdcard {
    non-removable;
    bus-width = <8>;
    mmc-ddr-1_8v;
    mmc-hs200-1_8v;
    mmc-hs400-1_8v;
    no-sdio;
-   no-sd;
+   //no-sd;
    ctl-spec-caps = <0x308>;
    cap-mmc-highspeed;
    sunxi-power-save-mode;
    sunxi-dis-signal-vol-sw;
    max-frequency = <100000000>;
    vmmc-supply = <&reg_dcdc1>;
    /*emmc io vol 3.3v*/
    vqmmc-supply = <&reg_bldo1>;
    /*emmc io vol 1.8v*/
    /*vqmmc-supply = <&reg_eldo1>;*/
    status = "disabled";
};
```

1.5.2.4.3 内核配置

```
make kernel_menuconfig 层层选中
[*] Enable the block layer --->
  [*] Support for large (2TB+) block devices and files
  [*] Block layer SG support v4
Device Drivers --->
  < > Memory Technology Device (MTD) support ---> //取消勾选
```

```
[*] Block devices ---> //确认勾选
File systems --->
<*> The Extended 4 (ext4) filesystem
```

1.5.2.4.4 TINA 环境配置

```
make menuconfig 层层选中
Target Images --->
  Boot (SD Card) Kernel format (boot.img) ---> //默认选中
  [ ] For storage less than 32M, enable this when using ota //取消勾选

Global build settings --->
  [ ] Strip unnecessary functions from libraries //取消勾选

Utilities --->
  Filesystem --->
    <*> e2fsprogs //选中
```

1.5.3 是否支持 SPI Slave 模式

支持，驱动位于lichee/linux-4.9/drivers/spi/spi-sunxi.c

1.6 NPU 相关

1.6.1 NPU 问题：NPU 是否支持算子级别的接口

当前只支持模型级别，尚不支持算子级别。

1.6.2 NPU 的资料如何获取，是否有 License 问题

资料获取：一号通->文档中心->V85X->V85X IPC->Tool工具类文档->工具使用指南 需要通过邮件申请 License，具体参照一号通->文档中心->V85X->V85X IPC->Software软件类文档->基础组件开发指南->《Tina_Linux_NPU开发简介.pdf》和《Tina_Linux_NPU部署工具安装指导.pdf》

1.7 编码相关

1.7.1 编解码相关问题

待补充。

1.8 Camera&&ISP 相关

1.8.1 双摄像头如何配置

待补充。

1.8.2 ISP 设置区域曝光

调用接口 AW_MPI_ISP_SetLocalExposureArea()

1.9 显示相关

1.9.1 如何替换开机 Logo

准备好待替换的 bmp 图片，重命名成 bootlogo.fex 替换

`${Tina_SDK}/device/config/chips/v853/configs/perf1/linux/bootlogo.fex`

1.9.2 如何确认 LCD 屏本身工作是否正常

可打开 LCD 屏自测模式，测试 colorbar 输出是否正常。命令：`echo 1 > /sys/class/disp/disp/attr/colorbar`

1.10 GPIO 类（驱动能力设置）

1.10.1 芯片内部的 GPIO 是否带上下拉电阻

芯片设计时，GPIO 是有带上下拉电阻，约为 100K，精度为 20%.

1.10.2 GPIO 的驱动能力如何配置

1.10.2.1 pinctrl 调试

1.10.2.1.1 查看引脚信息

```
mount -t debugfs none /sys/kernel/debugfs
cd /sys/kernel/debug/sunxi_pinctrl
```

查看命令：

```
echo PIN_NAME > sunxi_pin (例: echo PB2 > sunxi_pin)
cat sunxi_pin_configure
```

其中PIN_NAME是由(pin-group + pin-number)组成, pin-group为A, B, ...G的分组 ; pin-number为0, 1, ...9的标号, 以驱动中的B组第2号引脚为例, pin-name为: PB2。

1.10.2.1.2 更改引脚属性 每个引脚都有四种属性:

data	电平
dlevel	驱动能力
pull	上拉/下拉
function	功能选择

更改属性命令: echo PIN_NAME value > 属性如果改变 PB2 引脚的 data 属性为 1. echo PB2 1 > data 其他属性操作类似相关属性的默认值由 sys_config 或 dts 配置确定;

```
/sys/kernel/debug/sunxi_pinctrl #
/sys/kernel/debug/sunxi_pinctrl # echo PB2 > sunxi_pin
/sys/kernel/debug/sunxi_pinctrl # cat sunxi_pin_configure
pin[PB2] funciton: 7
pin[PB2] data: 0
pin[PB2] dlevel: 1
pin[PB2] pull: 0
/sys/kernel/debug/sunxi_pinctrl # echo PB2 1 > function
/sys/kernel/debug/sunxi_pinctrl # echo PB2 1 > data
/sys/kernel/debug/sunxi_pinctrl # echo PB2 2 > dlevel
/sys/kernel/debug/sunxi_pinctrl # echo PB2 1 > pull
/sys/kernel/debug/sunxi_pinctrl # cat sunxi_pin_configure
pin[PB2] funciton: 1
pin[PB2] data: 1
pin[PB2] dlevel: 2
pin[PB2] pull: 1
/sys/kernel/debug/sunxi_pinctrl #
```

图 1-8: sunxi-pinctrl

1.10.2.2 Linux 标准 GPIO 调试

确认系统中有 “/sys/class/gpio” 这级目录, 如果没有, 在编译内核的时候加入

Device Drivers

- GPIO Support
- /sys/class/gpio/... (sysfs interface)

/sys/class/gpio使用说明:

1. 通过/sys/文件接口操作IO端口，即GPIO到文件系统的映射；
2. 控制GPIO的目录位于/sys/class/gpio；
3. /sys/class/gpio/export文件，用于通知系统需要导出控制的GPIO引脚编号；
4. /sys/class/gpio/unexport文件，用于通知系统取消导出；
5. /sys/class/gpio/gpiochipX目录保存系统中GPIO寄存器的信息，包括每个寄存器控制引脚的起始编号base，寄存器名称，引脚总数 导出一个引脚的操作步骤

- 首先计算此引脚编号 引脚编号 = 控制引脚的寄存器基数 + 控制引脚寄存器位数举例（具体GPIO 参考数据手册），如想控制 PB2 引脚，那么引脚编号就等于 $1 \times 32 + 2 = 34$ ；
- export 引脚向/sys/class/gpio/export 写入此编号，比如 34 号引脚，在 shell 中可以通过以下命令实现：echo 34 > /sys/class/gpio/export 命令成功后生成 /sys/class/gpio/gpio34 目录，如果没有出现相应目录，说明此引脚不可导出。
- 定义输入输出 direction 文件，定义输入输出方向，可以通过下面命令定义为输出。echo out > /sys/class/gpio/gpio34/direction direction 接受的参数可以是：in、out、high、low。其中参数 high / low 在设置方向为输出的同时，将 value 设置为相应的 1 / 0。
- 设置 value 值 value 文件是端口的数值，为 1 或 0，通过下面命令将 gpio34 设置为高电平。echo 1 > /sys/class/gpio/gpio34/value

1.11 PMU 相关

1.11.1 PMU 调试命令

- virtual regulator 调试节点

此节点为直接控制调节各路电压的输出，使用方法如下，shell 命令设置 regulator 电压：

shell命令设置regulator电压。

virtual设备存在于axp2101主设备结点下面，因此设备路径为主设备下面的从设备。以AXP2101的设备举例。
/sys/devices/platform/soc/twi4/i2c-4/4-0034/regulator/regulator.1/reg-virt-consumer.1-dcdc1 /
通过此路径下面的max_microvolts和min_microvolts设备结点进行写操作，用来完成对设备电源的控制，此例为：
echo 3000000 > max_microvolts
echo 3000000 > min_microvolts
设置电压为3000000uV，3000mV，3V。

- 查看设备引用 regulator

```
root@none):/sys/kernel/debug/regulator# cat regulator_summary
regulator          use open bypass voltage current    min    max
-----
regulator-dummy    0    4    0    0mV    0mA    0mV    0mV
  uart0            0mV    0mV
  twi4              0mV    0mV
  twi1              0mV    0mV
  twi0              0mV    0mV
axp2101-dcdc1       0    5    0 3300mV  0mA 1500mV 3400mV
  sdc0              0mV    0mV
  sdc0              0mV    0mV
  sdc0              0mV    0mV
  reg-virt-consumer.1 0mV    0mV
  spi0              0mV    0mV
axp2101-dcdc2       0    1    0  900mV  0mA  500mV 1540mV
  reg-virt-consumer.2 0mV    0mV
axp2101-dcdc3       0    2    0  900mV  0mA  500mV 3400mV
  cpu0              900mV  900mV
  reg-virt-consumer.3 0mV    0mV
axp2101-dcdc4       0    1    0 1500mV  0mA  500mV 1840mV
  reg-virt-consumer.4 0mV    0mV
axp2101-dcdc5       0    2    0 1200mV  0mA 1200mV 3700mV
  sensor0           1200mV 3300mV
  reg-virt-consumer.5 0mV    0mV
axp2101-rtcldo      0    0    0 1800mV  0mA 1800mV 1800mV
axp2101-rtcldo1     0    0    0 1800mV  0mA 1800mV 1800mV
axp2101-aldol       0    1    0 1800mV  0mA  500mV 3500mV
```

- regmap registers 打印 PMU 的所有的寄存器值
- 需要先挂载 debugfs 节点，shell 操作命令如下：

```
mount -t debugfs none /sys/kernel/debug
```

- 写寄存器操作，往寄存器 0xff 写入 0x01

```
echo 0xff 0x01 > /sys/kernel/debug/regmap/4-0034/registers
```

- 读寄存器，读取所有的 PMIC 的寄存器值

```
cat /sys/kernel/debug/regmap/4-0034/registers
```

- 通过/sys/class/axp 节点修改 PMIC 寄存器的值
- 读寄存器值：

```
echo 0x68 > /sys/class/axp/axp_reg;
cat /sys/class/axp/axp_reg
```

- 写寄存器：reg:0x68 写 0x01

```
echo 0x6801 > /sys/class/axp/axp_reg
```

- sys_config.fex 修改电压只对 uboot 阶段起作用，到 kernel 阶段有的模块会修改电压。

1.12 硬件信号测试相关

1.12.1 wifi 相关问题：XR819/XR829 调大发射功率

为了满足某些测试需求，需要对最大发射功率进行调整，但强烈建议不要輕易的修改发射功率，否则可能导致性能较差或者无法通过认证机构的认证。具体修改方法可使用 SddEditor（一号通可获取）工具进行修改，操作步骤如下图 xr829-0，打开 sdd_xr829.bin 文件，点击 Min/Max Power For Modulations，会弹出图 xr829-1 的窗口，根据需要修改的各速率的发射功率后点击 OK 按钮返回第一级界面，继续点击 OK 按钮保存修改到 sdd_xr829.bin 文件，然后把 sdd_xr829.bin 文件更新到设备即可。

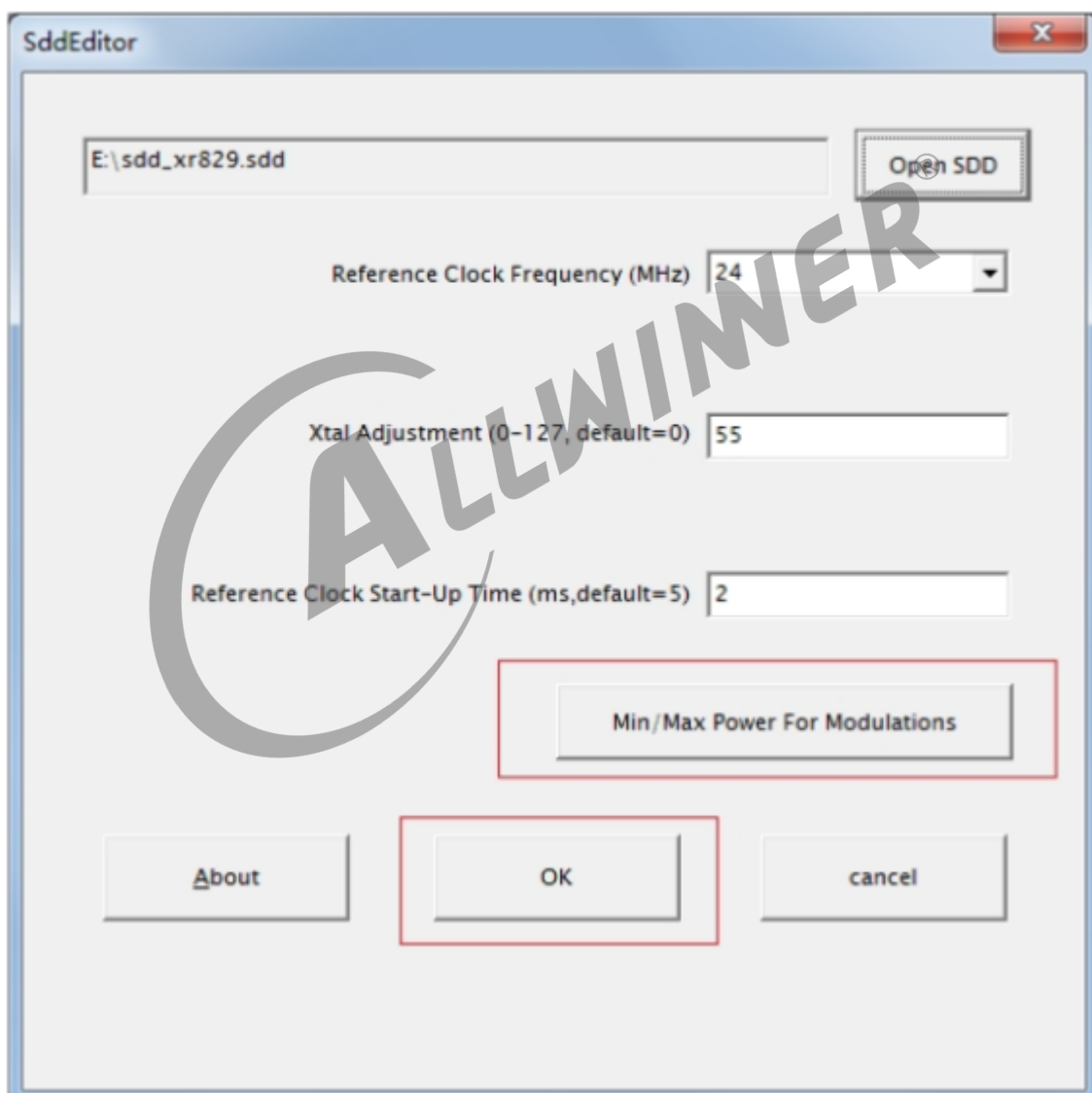


图 1-9: xr829-0

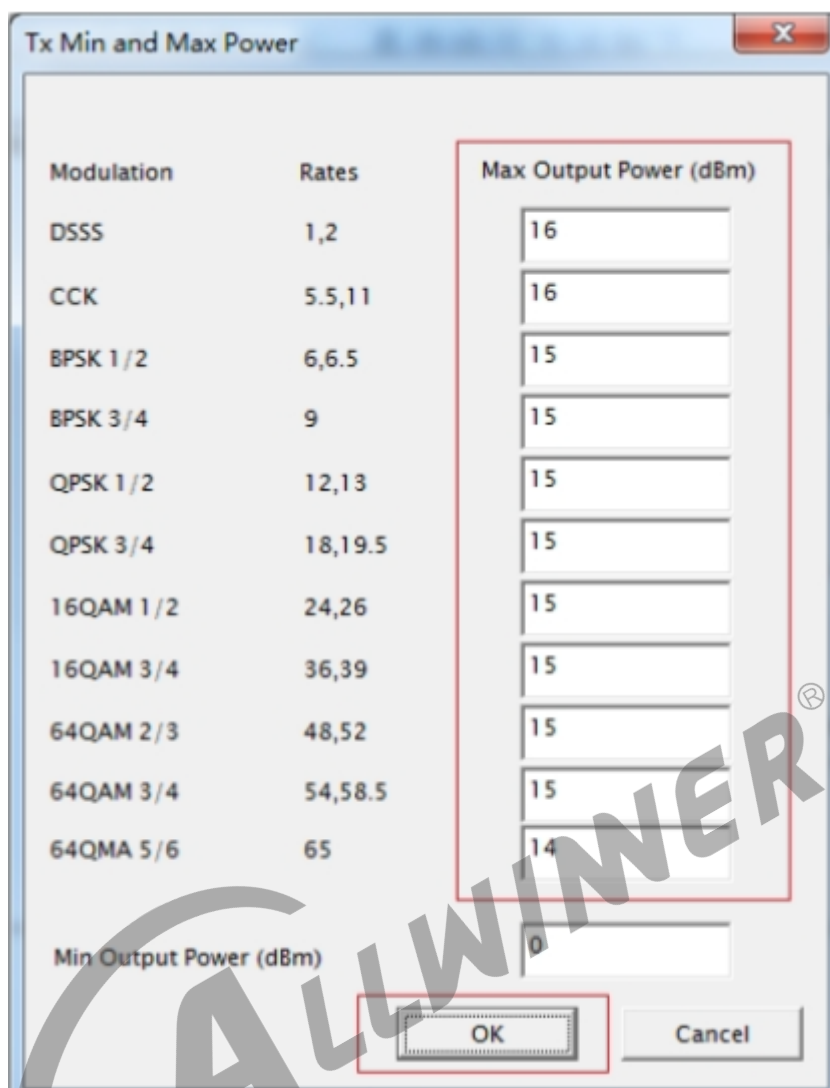


图 1-10: xr829-1

1.12.2 如何将 WIFI 的 MAC 地址固件下来 (XR819/XR829)

XR 系列 wifi 芯片：通过 DragonSN 烧号工具，将 MAC 地址烧录到 Flash 上的 private 分区，系统挂载的时候将 private 分区挂载，然后读取 mac 地址，设置到系统中 Realtek 芯片：通过烧写 wifi 模组的 efuse 区域将 MAC 地址固定；

1.12.3 EMI 相关问题：SDIO 展频、mipi-csi 展频、LCD 展频

1.12.3.1 DRAM 展频操作

- 通过设置 dram 的参数 dram_tpr13 相应的位数即可，如下：

展频开关使能由bit23控制：
0:disable
1:enable

展频系数由bit[22:20]设置：
1-5：对应展频系数0.1-0.5
others:展频系数0.4

1.12.3.2 SDIO 展频

因 SDIO 接口的 PLL 只用 PLL_PERI，因为只要对 PLL_PERI 进行展频即可。

1.12.3.2.1 展频设计

- 配置 PLL_CTRL 寄存器：0x02001020 = 0xc9216310

PLL_CTRL 的寄存器为：0x02001020, 通过 echo 0x02001020 > /sys/class/sunxi_dump/dump && cat /sys/class/sunxi_dump/dump 读出寄存器的值

- 可通过设置 N 和 M1 的值来设置 PLL_PERI 的频率，不过目前频率已设置好。
 $PLL_PERI = 0x02001040 = 0xc9216310$ (N=bit[15:8]+1=100 M1=bit1+1=1 P0=bit[18:16]+1=2) $PLL_PERI = 24 * N / M1/P0 = 1200MHz$
- 配置 bit24: SDM_Enabe 为 1，使能展频功能
- 配置 PLL_Pattern 寄存器:0x02001120 = 0xd1303333

通过 echo 0x02001120 > /sys/class/sunxi_dump/dump && cat /sys/class/sunxi_dump/dump

此寄存器是用来对时钟进行展频幅度调整，涉及两个参数 SDM_bot 和 Wave_step 的值，计算如下：

比如展频设置为 1.6%，即上下展频 0.8%，范围为：1190.4MHz(360 * 0.992) ~ 1209.6MHz(368 * 1.008)

$X1 = (1190.4 - (N*24/M1/P0)) / 24 = -0.4$ (负数不符合展频范围, 取 0 即可)

$x2 = (1209.6 - (N*24/M1/p0)) / 24 = 0.4$

- 展频基频选择由 PLL_Pattern 寄存器的 bit[18:17] 位选择, FREQ 默认选择 31.5KHz 即可
- PLL_Pattern 寄存器寄存器的值计算:

$$\text{SMD_bot}[16:0] = 2^{17} * X1 = 0$$

$$\text{Wave_step}[28:20] = 2^{17} * (X2 - X1) / (24\text{MHz}/\text{FREQ}) * 2 = 0x8A$$

$$\text{PLL_Pattern} = 0xC8600000$$
- 通过串口直接写寄存器开启展频

```
/* 有的模块可能也直接使用了此PLL_PERI时钟源, 如果关闭了导致系统直接挂死, 此条命令可不写 */
echo 0x02001020 0xc0216310 > /sys/class/sunxi_dump/write; /* 先disable PLL_OUTPUT[bit30] 0:
disable 1:enable
* 和PLL_SDM[bit24] 0:disable 1:
enable*/
echo 0x02001120 0xC8600000 > /sys/class/sunxi_dump/write; /*设置pattern0*/
/* 第一条命令不写时, 此条命令也不写 */
echo 0x02001020 0xc1216310 > /sys/class/sunxi_dump/write; /*enable PLL_SDM_ENABLE展频使能位*/
echo 0x02001020 0xc9216310 > /sys/class/sunxi_dump/write; /*enable PLL_OUTPUT*/
```

1.12.3.3 mipi-csi 展频

csi 的展频可参考 SDIO 或者 LCD, 本意有可能是针对 sensor, 也有可能是针对 VIN 本身模块, 都是要找到对应的 PLL 时钟, 然后对齐进行时钟展频, 如对 csi_top 时钟进行展频, 先查看 CSI_TOP 挂载哪路 PLL 时钟:

```
mount -t debugfs none /sys/kernel/debugfs
cat /sys/kernel/debug/clk/clk_summary
.....
pll_periph0x2          1          1 1200000000          0 0
  pll_periph0400m      0          0 400000000          0 0
    ce                 0          0 400000000          0 0
      pll_periph0200m  0          0 200000000          0 0
  pll_periph0600m      3          3 600000000          0 0
    apb0               5          5 100000000          0 0
      fanout_pclk      0          0 100000000          0 0
        wiegand        1          1 100000000          0 0
          ths           1          1 100000000          0 0
            pwm         1          1 100000000          0 0
  ahb                  6          6 200000000          0 0
    dpss_top           1          1 200000000          0 0
      usbotg           1          1 200000000          0 0
        usbehci0       0          0 200000000          0 0
          usbohci0     0          0 200000000          0 0
            gmac        0          0 200000000          0 0
              iommu     1          1 200000000          0 0
                hstimer 0          0 200000000          0 0
                  spinlock 1          1 200000000          0 0
                    msgbox1 0          0 200000000          0 0
                      msgbox0 1          1 200000000          0 0
                        dma   1          1 200000000          0 0
  pll_periph0300m      3          4 300000000          0 0
    csi_top
```

可以看出来 csi_top 的时钟是挂到了 pll_periph0x2—>pll_periph0300m—>csi_top 的时钟上面，只要对进行 pll_periph0x2 展频即可，也即对 PLL_PEREI 进行展频。

直接参考 SDIO 的展频即可，如果挂载其他的 PLL 上，如 PLL_VIDEO0/PLL_CSI 等，按照相同的原理去计算，然后展频即可。

1.12.3.4 mipi-dsi 展频 && CPU 接口的 LCD 展频

不管是 MIPI 接口还是 CPU 接口，其原理都是对父时钟进行展频，如 V85x-dsi，在屏幕点亮的情况下，可通过此命令查看屏幕的 clk 关系，如下：

```
mount -t debugfs none /sys/kerner/debugfs
cat /sys/kernel/debug/clk/clk_summary
.....
    pll_video0x4          1          2  360000000          0 0
      tcon_lcd            1          1  360000000          0 0
    pll_video0            0          1   90000000          0 0
    pll_video0x2          0          0  180000000          0 0
.....
```

可以看出来 tcon_lcd 的时钟是挂到了 pll_video0x4 的时钟上面，只要对 pll_video0x4 进行展频即可，因为 pll_video0 和 pll_video0x2 是由 pll_video0x4 经过 4 分频和 2 分频得到，在对 pll_video0x4 进行展频是要考虑使用了 video0 和 video0x2 为时钟源的模块是否有影响。

1.12.3.4.1 展频设计

- 配置 PLL_CTRL 寄存器: 0x02001040 = 0xc9001d03

pll_video0x4 的寄存器为: 0x02001040, 通过 echo 0x02001040 > /sys/class/sunxi_dump/dump && cat /sys/class/sunxi_dump/dump 读出寄存器的值

- 可通过设置 N 和 M1 的值来设置 pll_video0x4 的频率，不过目前频率已设置好。

VIDEOE0_PLL_REG = 0x02001040 = 0xc9001d03 (N=0x1d+1=30 M1=bit1+1=2 M0=bit0+1=2) $\text{pll_video0x4} = 24 * N / M1 = 360\text{MHz}$

- 配置 bit24: SDM_Enabe 为 1，使能展频功能
- 配置 PLL_Pattern 寄存器: 0x02001140 = 0xd1303333

通过 echo 0x02001140 > /sys/class/sunxi_dump/dump && cat /sys/class/sunxi_dump/dump

此寄存器是用来对时钟进行展频幅度调整，涉及两个参数 SDM_bot 和 Wave_step 的值，计算如下：

比如展频设置为 1.6%，即上下展频 0.8%，范围为：357.12(360 * 0.992) ~ 362.88MHz(368 * 1.008)

$X1 = (357.12 - (N \times 24 / M1)) / 24 = -0.12$ (负数不符合展频范围, 取 0 即可)

$x2 = (362.88 - (N \times 24 / M1)) / 24 = 0.12$

- 展频基频选择由 PLL_Pattern 寄存器的 bit[18:17] 位选择, FREQ 默认选择 31.5KHz 即可
- PLL_Pattern 寄存器寄存器的值计算:

$SMD_bot[16:0] = 2^{17} \times X1 = 0$ $Wave_step[28:20] = 2^{17} \times (X2 - X1) / (24MHz / FREQ)$
 $\times 2 = 0x29$

PLL_Pattern = 0xC2900000

- 通过串口直接写寄存器开启展频

```
echo 0x02001040 0xc0001d03 > /sys/class/sunxi_dump/write; /* 先disable PLL_OUTPUT[bit30] 0:
disable 1:enable
                                * 和PLL_SDM[bit24] 0:disable 1:
enable*/
echo 0x02001140 0xc2900000 > /sys/class/sunxi_dump/write; /*设置pattern0*/ ②
echo 0x02001040 0xc1001d03 > /sys/class/sunxi_dump/write; /*enable PLL_SDM_ENABLE展频使能位*/
echo 0x02001040 0xc9001d03 > /sys/class/sunxi_dump/write; /*enable PLL_OUTPUT*/
```

1.12.4 如何进入 USB 眼图测试

1.12.4.1 USB0 OTG Device 眼图测试:

手动切换Device模式:
`cat /sys/devices/platform/soc/usbc0/usb_device`

每个芯片平台的节点路径会有些差异, 请通过find命令确认:
`find /sys -name otg_ed_test`

眼图测试命令:
`echo test_pack > /sys/devices/platform/soc/5100000.udc-controller/otg_ed_test`

1.12.4.2 USB0 OTG Host 眼图测试:

手动切换Host模式:
`cat /sys/devices/platform/soc/usbc0/usb_host`

每个芯片平台的节点路径会有些差异, 请通过find命令确认(注意选择带有ehci0的路径):
`find /sys -name ed_test`

眼图测试命令:
`echo test_pack > /sys/devices/platform/soc/5101000.ehci0-controller/ed_test`

1.13 工具类

1.13.1 如何获取量产工具二次开发 SDK 包

说明：当前支持二次开发的工具包括 LiveProc 二次开发 SDK、DragonSN 二次开发 SDK、DragonMAT 二次开发 SDK 和 USBProproc 开发 SDK。

申请流程 1、客户端通过 Aservice 发起工具申请。2、FAE 或产品经理在一号通->工具管理->量产工具二次开发SDK管理操作外发申请。



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。