目录

**[1. BIOS编译](#_Toc1332799135_WPSOffice_Level1)** **[1](#_Toc1332799135_WPSOffice_Level1)**

[1.1 pmon编译及注意事项](#_Toc695600833_WPSOffice_Level2) [1](#_Toc695600833_WPSOffice_Level2)

**[2. 内核编译](#_Toc695600833_WPSOffice_Level1)** **[4](#_Toc695600833_WPSOffice_Level1)**

[2.1 编译环境及依赖工具下载](#_Toc1060235000_WPSOffice_Level2) [4](#_Toc1060235000_WPSOffice_Level2)

[2.2 指定交叉编译工具链](#_Toc288954644_WPSOffice_Level2) [4](#_Toc288954644_WPSOffice_Level2)

[2.3 设置内核配置文件](#_Toc1232742147_WPSOffice_Level2) [4](#_Toc1232742147_WPSOffice_Level2)

[2.4 打开图形化配置界面（可在此环节修改内核相关配置）](#_Toc283777930_WPSOffice_Level2) [4](#_Toc283777930_WPSOffice_Level2)

[2.5 编译内核](#_Toc977865982_WPSOffice_Level2) [5](#_Toc977865982_WPSOffice_Level2)

**[3. 文件系统制作](#_Toc1060235000_WPSOffice_Level1)** **[6](#_Toc1060235000_WPSOffice_Level1)**

[3.1 buildroot编译](#_Toc851935165_WPSOffice_Level2) [6](#_Toc851935165_WPSOffice_Level2)

[3.2 LoongOS编译](#_Toc49468056_WPSOffice_Level2) [6](#_Toc49468056_WPSOffice_Level2)

**[4. 启动与调试](#_Toc288954644_WPSOffice_Level1)** **[7](#_Toc288954644_WPSOffice_Level1)**

[4.1 启动](#_Toc1135452428_WPSOffice_Level2) [7](#_Toc1135452428_WPSOffice_Level2)

[4.2 ejtag调试](#_Toc1182147037_WPSOffice_Level2) [10](#_Toc1182147037_WPSOffice_Level2)

**[5. 接口配置与使用](#_Toc1232742147_WPSOffice_Level1)** **[12](#_Toc1232742147_WPSOffice_Level1)**

[5.1 GMAC](#_Toc350184606_WPSOffice_Level2) [12](#_Toc350184606_WPSOffice_Level2)

[5.2 VGA](#_Toc1220256492_WPSOffice_Level2) [14](#_Toc1220256492_WPSOffice_Level2)

[5.3 OTG](#_Toc386732039_WPSOffice_Level2) [16](#_Toc386732039_WPSOffice_Level2)

[5.4 USB2.0](#_Toc637388364_WPSOffice_Level2) [18](#_Toc637388364_WPSOffice_Level2)

[5.5 USB3.0](#_Toc1674743342_WPSOffice_Level2) [19](#_Toc1674743342_WPSOffice_Level2)

[5.6 PCIE](#_Toc209638542_WPSOffice_Level2) [20](#_Toc209638542_WPSOffice_Level2)

[5.7 PCI](#_Toc524684784_WPSOffice_Level2) [22](#_Toc524684784_WPSOffice_Level2)

[5.8 SDIO](#_Toc1150195336_WPSOffice_Level2) [25](#_Toc1150195336_WPSOffice_Level2)

[5.9 NAND](#_Toc183626816_WPSOffice_Level2) [27](#_Toc183626816_WPSOffice_Level2)

[5.10 CAN](#_Toc938961565_WPSOffice_Level2) [31](#_Toc938961565_WPSOffice_Level2)

[5.11 I2C](#_Toc594498478_WPSOffice_Level2) [34](#_Toc594498478_WPSOffice_Level2)

[5.12 SPI](#_Toc1758972982_WPSOffice_Level2) [41](#_Toc1758972982_WPSOffice_Level2)

[5.13 RS485/UART3](#_Toc1161023934_WPSOffice_Level2) [46](#_Toc1161023934_WPSOffice_Level2)

[5.14 WATCH DOG](#_Toc1816746239_WPSOffice_Level2) [48](#_Toc1816746239_WPSOffice_Level2)

[5.15 PWM](#_Toc252242434_WPSOffice_Level2) [50](#_Toc252242434_WPSOffice_Level2)

[5.16 GPIO中断](#_Toc641362929_WPSOffice_Level2) [53](#_Toc641362929_WPSOffice_Level2)

**[附录A： 2K500复用设置注意事项](#_Toc283777930_WPSOffice_Level1)** **[54](#_Toc283777930_WPSOffice_Level1)**

**[(1) 默认复用设置](#_Toc977865982_WPSOffice_Level1)** **[54](#_Toc977865982_WPSOffice_Level1)**

**[(2) cfg\_func\_multi函数设置](#_Toc851935165_WPSOffice_Level1)** **[54](#_Toc851935165_WPSOffice_Level1)**

**[(3) pmon命令行下调用命令修改复用配置](#_Toc49468056_WPSOffice_Level1)** **[55](#_Toc49468056_WPSOffice_Level1)**

**[(4) 复用冲突处理](#_Toc1135452428_WPSOffice_Level1)** **[55](#_Toc1135452428_WPSOffice_Level1)**

|  |  |
| --- | --- |
| 版本记录 | |
| 版本号：V0.1 | |
| V0.1 | 重新加上了编译环境搭建以及依赖工具下载的内容。 |
|  |  |

# BIOS编译

## 1.1 pmon编译及注意事项

以编译2K500的pmon源码为例：

### pmon源码

将源码压缩包拷贝到工作目录，使用下述命令对源码进行解压（注：如果使用的工作环境是虚拟机，请不要直接在共享文件夹下进行解压）

tar -zxvf pmon-ls2k500.tar.gz

### 解压交叉编译工具

来到根目录下，解压交叉编译工具，默认会放入opt目录下。

cd /

tar -zxvf loongarch\_toolchain.tar.gz

### 编译环境及依赖工具下载

编译环境：ubuntu20.04

1. sudo apt install aptitude

sudo aptitude install xutils-dev

1. sudo apt install bison flex

进入pmon源码下进行工具编译：

cd PMON源码/tools/pmoncfg

make pmoncfg

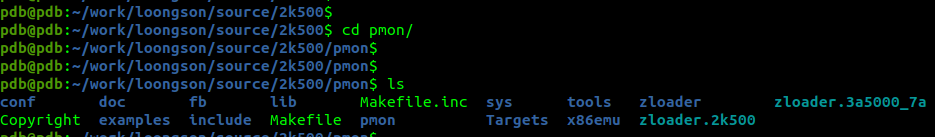
sudo cp pmoncfg /usr/bin

1. sudo apt install acpica-tools

### 根据下述命令编译pmon

解压过程如果没有“error”信息，则可继续进行下述步骤。

解压完成后，会在当前目录下看到一个pmon/目录，使用命令cd pmon/进入源码目录下，使用ls命令可看到如下目录结构。



#### 进入2K500对应的编译目录

cd zloader.2k500/

#### 设置环境变量，指定当前目录下的编译工具

export PATH=/opt/loongarch\_toolchain/bin:$PATH

注：编译工具链loongarch\_toolchain.tar.gz如果解压在/opt目录下，则使用上述命令进行设置，否则需修改为工作环境下编译工具所在的路径。

#### 使用命令编译BIOS：

make cfg all tgt=rom ARCH=loongarch CROSS\_COMPILE=loongarch64-linux-gnu- DEBUG=-g

编译选项解释：

make cfg 对pmon进行配置；

all为Makefile里的编译项；

tgt=rom，指定tgt为rom，则会生成gzrom.bin文件；

ARCH=loongarch，指定架构为loongarch；

CROSS\_COMPILE=loongarch64-linux-gnu-，指定编译工具前缀名；

DEBUG=-g，设置编译的时候携带调试信息。

#### 编译设备树，并生成携带设备树的BIOS文件”gzrom-dtb.bin”

make dtb ARCH=loongarch CROSS\_COMPILE=loongarch64-linux-gnu-

编译完成后，会在zloader.2k500/下，看到最终生成的携带设备树的gzrom-dtb.bin文件，以及不带设备树的gzrom.bin文件。

注：如更改了配置文件 Targets/LS2K500/conf/ls2k500, 则在编译前要执行make cfg，使得更改生效，如果普通编译没有更改配置，则每次无需都执行 make cfg 命令。

### pmon编译脚本文件

可在pmon源码目录下，创建下述脚本文件，如需重新编译pmon，则只需执行该脚本即可。

#!/bin/bash

cd zloader.2k500/

export PATH=/opt/loongarch\_toolchain/bin:$PATH

make cfg all tgt=rom ARCH=loongarch CROSS\_COMPILE=loongarch64-linux-gnu- DEBUG=-g

make dtb ARCH=loongarch CROSS\_COMPILE=loongarch64-linux-gnu-

# 内核编译

## 编译环境及依赖工具下载

编译环境：ubuntu20.04

sudo apt install libncurses5‐dev

sudo apt install libssl-dev

## 指定交叉编译工具链

export PATH=/opt/loongarch\_toolchain/bin:$PATH

## 设置内核配置文件

2K500默认配置文件为arch/loongarch/configs/loongson3\_defconfig，需将其拷贝为.config文件。

cp arch/loongarch/configs/loongson3\_defconfig .config

## 打开图形化配置界面（可在此环节修改内核相关配置）

make menuconfig ARCH=loongarch

注：执行完步骤2.2之后，必须需执行步骤2.3，弹出图形界面后，选择<Exit>，然后保存退出。（默认配置文件中存在一些依赖配置选项，打开图形化界面并保存后，会自动勾选上这些依赖配置进行编译）

## 编译内核

make vmlinuz ARCH=loongarch CROSS\_COMPILE=loongarch64-linux-gnu-

编译完成后，会在当前目录下看到生成的vmlinux文件，与压缩后的内核文件vmlinuz。

# 文件系统制作

## buildroot编译

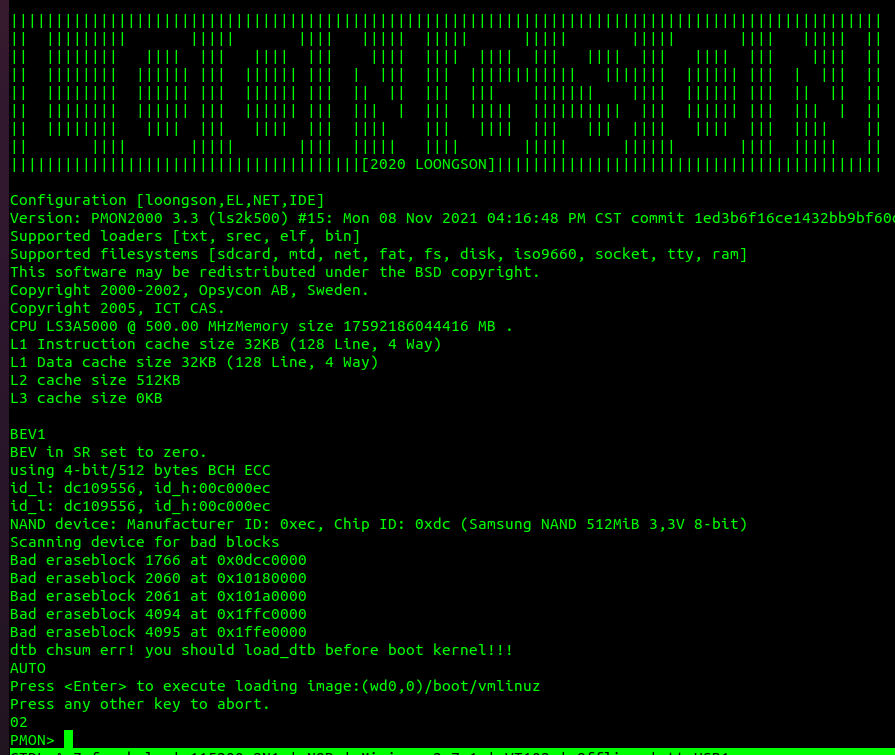
## 3.2 LoongOS编译

# 启动与调试

## 启动

### pmon启动与烧写

2K500支持多种启动方式，默认使用的为SPI启动。板卡拿到手，接上调试串口，就能从调试串口中看到pmon启动日志。在启动过程中按”c”，会打断pmon的自启动，并进入pmon命令行下,如下图。此时可通过下述方式进行在线更新pmon。



注：如果是板卡内norflash无pmon程序，或板卡内pmon无法正常启动，则可通过4.2小节，使用ejtag进行烧写。

#### u盘在线更新pmon

注：u盘需在板卡上电之前插上。

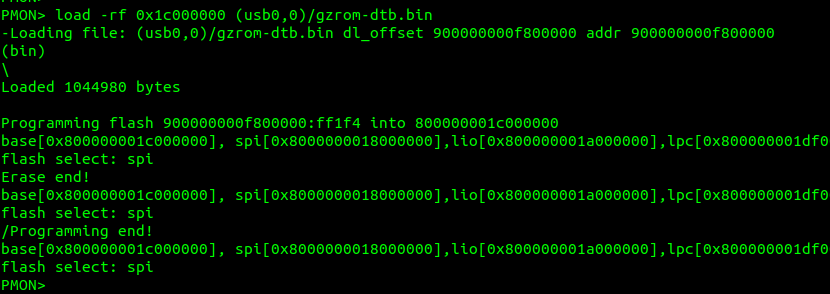
准备一个u盘，u盘最好在linux系统下进行格式化，且只有一个分区，分区号为0，若分区为fat格式，则将gzrom-dtb.bin文件放入u盘，且在板卡上电之前插上u盘，输入下述命令进行更新pmon。

load -rf 0x1c000000 (usb0,0)/gzrom-dtb.bin

(0x1c000000是flash设备映射的地址，usb0为u盘设备名，0为u盘第0个分区)

或者输入：load -rf 0x1c000000 [/dev/fs/fat@usb0/gzrom-dtb.bin](mailto:/dev/fs/fat@usb0/gzrom-dtb.bin)

如果u盘为ext2/ext3/ext4格式，则需将上述命令中fat替换成ext2进行加载。



#### 网口在线更新pmon

注：在主机端需搭建并启动tftp服务。

ifaddr syn0 板卡ip

load -rf 0x1c000000 tftp://主机ip/gzrom-dtb.bin

### 内核烧写及启动

#### 写入到内存启动，掉电数据丢失

在pmon启动过程中，按”c”进入pmon命令行后，可通过下述命令加载内核到内存中，并跳转启动内核。

##### 通过u盘加载并启动内核

注：u盘需在板卡上电之前插上。

u盘要求同4.1章节中，<u盘在线更新pmon>小节一致。

load (usb0,0)/vmlinuz

initrd (usb0,0)/rootfs.cpio.gz

g console=ttyS0,115200 rdinit=/sbin/init

##### 通过网口加载并启动内核

注：在主机端需搭建并启动tftp服务。

ifaddr syn0 板卡ip

load tftp://主机ip/vmlinuz

initrd tftp://主机ip/rootfs.cpio.gz

g console=ttyS0,115200 rdinit=/sbin/init

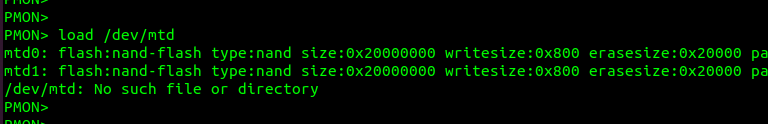
load命令加载内核到内存中；initrd命令加载文件系统到内存中；g命令进行跳转，并指定日志从ttyS0输出，内核启动后运行的第一个进程为/sbin/init。如果内核未与文件系统编译在一起，则执行上述步骤即可。如果内核与文件系统编译在了一起，则initrd命令需去掉。

#### 写入nand启动，掉电不丢失

在pmon启动过程中，按”c”进入pmon命令行后，可通过下述命令将程序烧写到nand中，设置自启动命令自动跳转内核。

查看pmon下nand分区情况：

load /dev/mtd



如上图，可见nand在pmon下分成了两个分区，则可在分区0中存放内核，分区1中存放yaffs2文件系统镜像，命令如下：

mtd\_erase /dev/mtd0

mtd\_erase /dev/mtd1

devcp tftp://主机ip/vmlinuz /dev/mtd0

devcp tftp://主机ip/rootfs-yaffs2.img /dev/mtd1y

set al /dev/mtd0

set append “console=ttyS0,115200 rw root=/dev/mtdblock1 rootfstype=yaffs2”

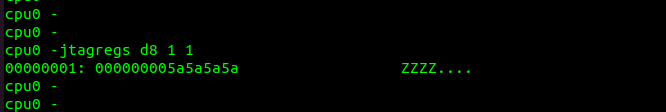
## ejtag调试

### ejtag烧写pmon

在工作目录下解压ejtag-debug.tar.gz压缩包，会在当前目录下看到ejtag-debug目录。将编译好的pmon二进制文件“gzrom-dtb.bin”放入ejtag-debug/目录下。

进入ejtag-debug/目录后，在终端执行sudo ./la\_dbg\_tool\_usb -t会进入ejtag命令行。

板卡上电后，输入：jtagregs d8 1 1，如果显示000000005a5a5a5a，则表示ejtag连接成功。



连接成功后，在ejtag命令行按照下述流程进行烧写：

板卡断电，

source configs/config.ls2k500

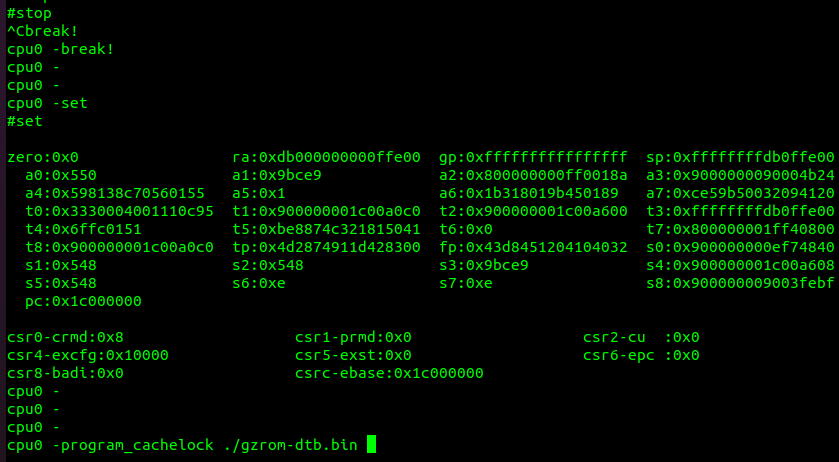
loop -1 stop

板卡上电，在ejtag命令行下通过“ctrl + c”打断，再输入下一条命令，

set

此时，能看到pc值指向0x1c000000，

program\_cachelock ./gzrom-dtb.bin



# 接口配置与使用

## GMAC

2K500-pai上，网口0为靠近USB3.0侧的网口；网口1为靠近VGA侧的网口。pmon命令行下通过devls命令可查看到网口设备名，分别为syn0与syn1。pmon与内核下网口默认已打开。

### pmon下设置网口ip

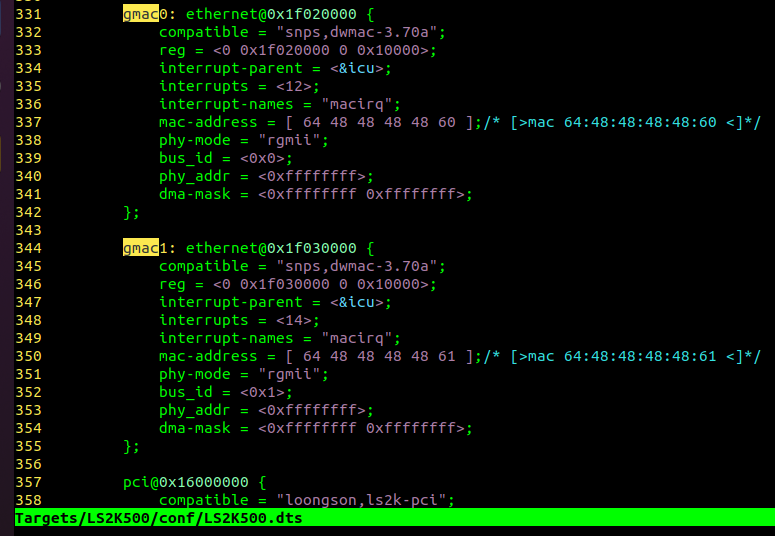
可通过ifaddr命令或者ifconfig命令在pmon命令行下设置网口ip。

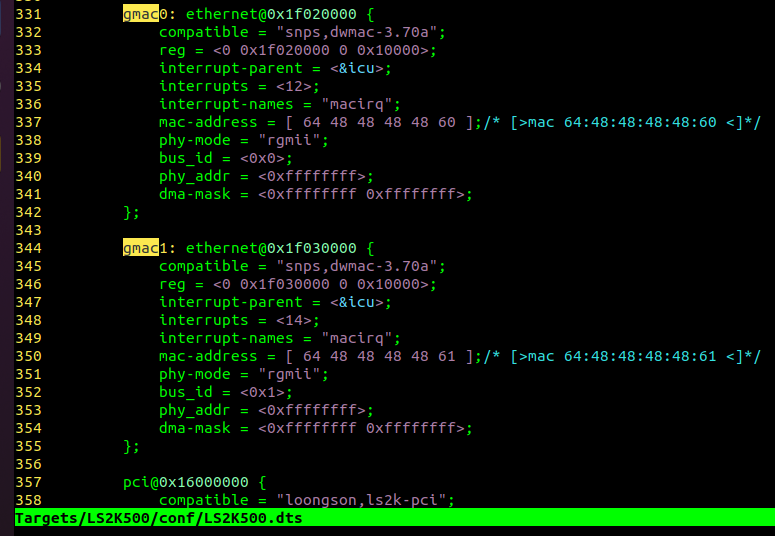
ifaddr syn0 板卡ip

或者ifconfig syn0 板卡ip

### pmon设备树中GMAC节点

Targets/LS2K500/conf/LS2K500.dts中打开两个gmac网口的节点。





### 内核GMAC驱动配置

CONFIG\_STMMAC\_ETH、CONFIG\_STMMAC\_PLATFORM。

### 板卡上网口使用的phy芯片为YT8511

pmon源码中sys/dev/gmac/synopGMAC\_network\_interface.c为网口驱动，驱动中init\_phy函数有YT8511 phy芯片的初始化操作。如果更换了其他phy芯片，可在该函数内对应phy芯片的读写延时进行初始化。

## VGA

### pmon下配置分辨率

在配置文件Targets/LS2K500/conf/ls2k500中打开1280x1024分辨率显示配置。



### pmon设备树中DC节点

Targets/LS2K500/conf/LS2K500.dts中打开dc显示节点。



### 内核DC驱动配置

CONFIG\_DRM\_LOONGSON\_VGA。

## OTG

pmon下无otg驱动，接口不可使用。内核下可设置主/从模式分别使用。otg为板卡上红框圈出来的接口。



### pmon设备树中OTG节点

设备树Targets/LS2K500/conf/LS2K500.dts里需打开otg节点。



### 内核下OTG主模式配置

CONFIG\_USB\_DWC2\_HOST。

### 内核下OTG从模式配置

（示例中模拟成打印设备，otg从模式同时只可作为一种从设备）

打开下述配置：

CONFIG\_USB\_OTG\_WHITELIST、

CONFIG\_USB\_DWC2\_PERIPHERAL、 CONFIG\_USB\_G\_PRINTER；

关闭下述配置：

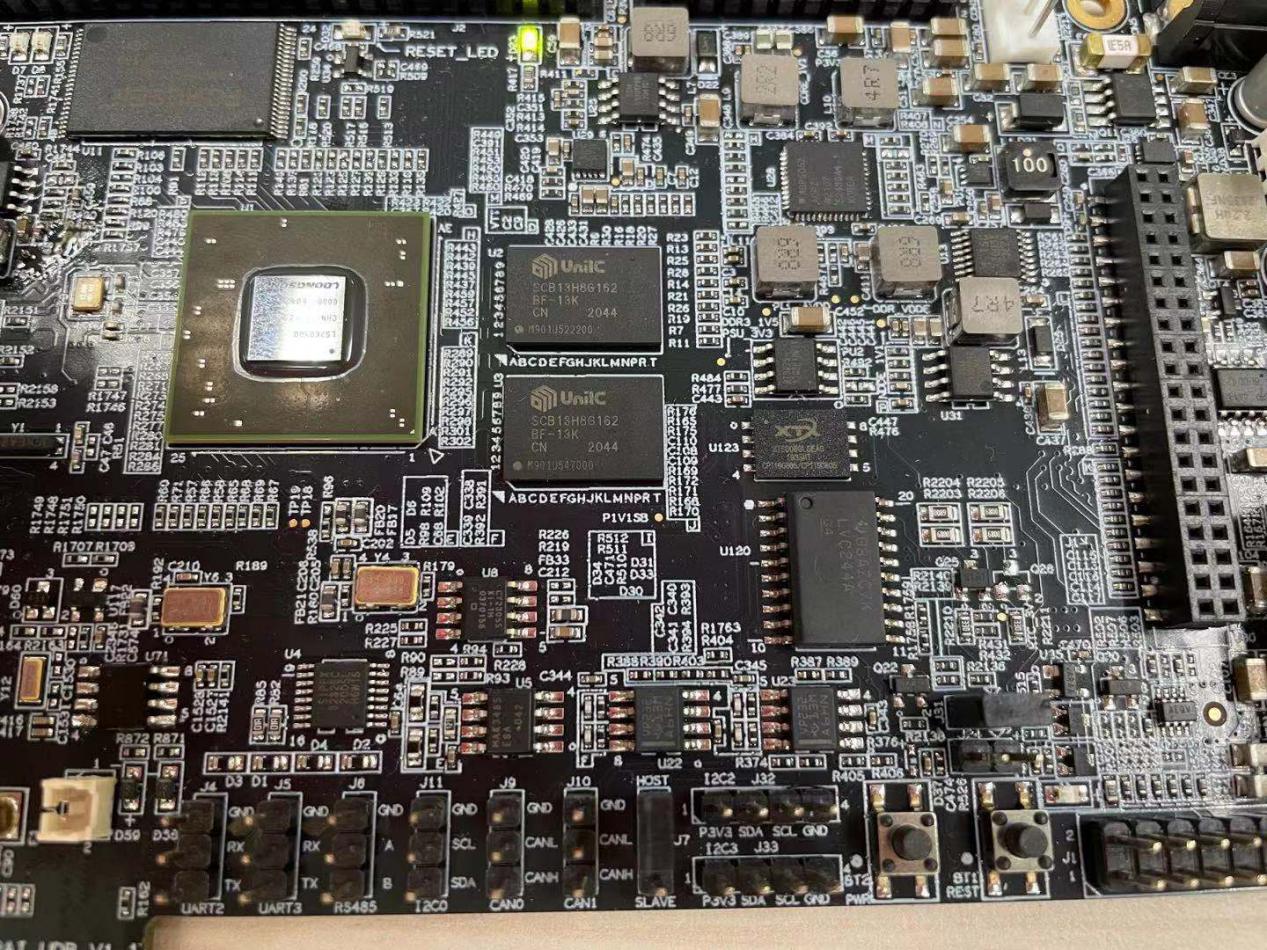
CONFIG\_USB\_CONFIGFS。

### 内核下OTG双模式配置

CONFIG\_USB\_OTG\_WHITELIST、

CONFIG\_USB\_DWC2\_DUAL\_ROLE、CONFIG\_USB\_G\_PRINTER。

此时，需要硬件跳线设置ID脚电平值，来区分主从模式。（ID脚拉低设置为主模式；ID脚拉高设置为从模式）  
 can1右侧的一排插针J7，短接上面两个引脚为HOST主模式，短接下面两个引脚为SLAVE从模式。（下图中短接成了HOST模式）

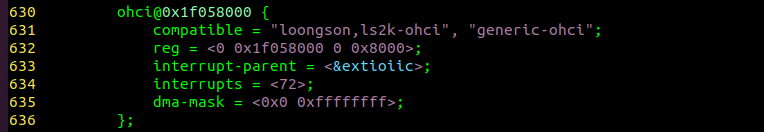


## USB2.0

### OHCI（低速/全速设备）

#### pmon设备树中OHCI节点

设备树Targets/LS2K500/conf/LS2K500.dts里需打开ohci节点。



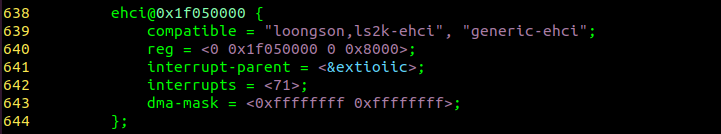
#### 内核下OHCI控制器驱动配置

CONFIG\_USB\_OHCI\_HCD\_PLATFORM。

### EHCI（高速设备）

#### pmon设备树中EHCI节点

设备树Targets/LS2K500/conf/LS2K500.dts里需打开ehci节点。



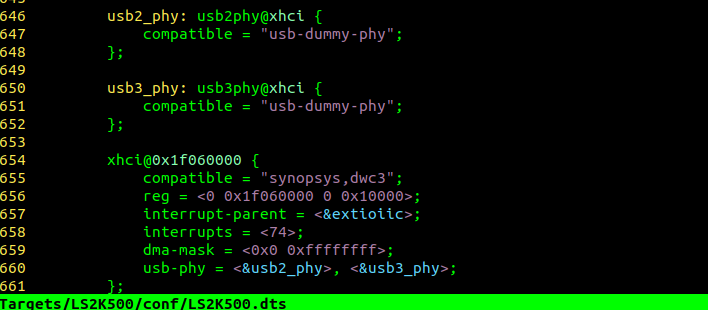
#### 内核下EHCI控制器驱动配置

CONFIG\_USB\_EHCI\_HCD\_PLATFORM。

## USB3.0

### pmon设备树中XHCI节点

设备树Targets/LS2K500/conf/LS2K500.dts里需打开xhci节点。



### 内核下DWC3主模式配置

CONFIG\_USB\_XHCI\_PLATFORM、CONFIG\_USB\_DWC3\_HOST。

### 内核下DWC3从模式配置

打开下述配置：

CONFIG\_USB\_XHCI\_PLATFORM、CONFIG\_USB\_DWC3\_GADGET、

CONFIG\_USB\_G\_PRINTER、CONFIG\_USB\_OTG\_WHITELIST。

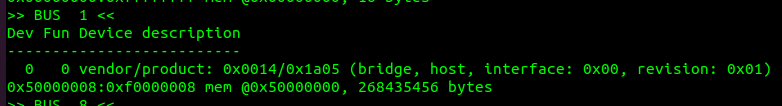
关闭下述配置：

CONFIG\_USB\_CONFIGFS。

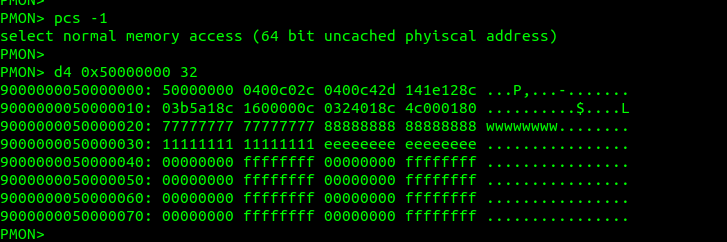
## PCIE

### PCIE0 EP模式使用示例

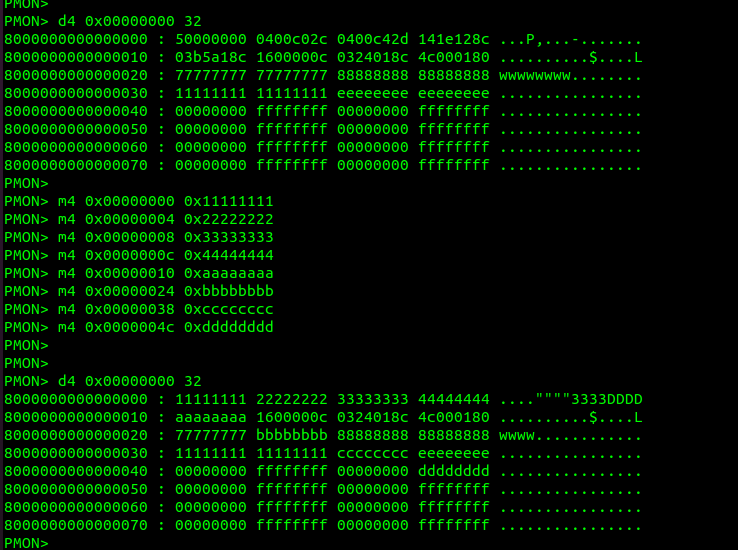
#### RC端pmon下，查看PCIE EP设备信息



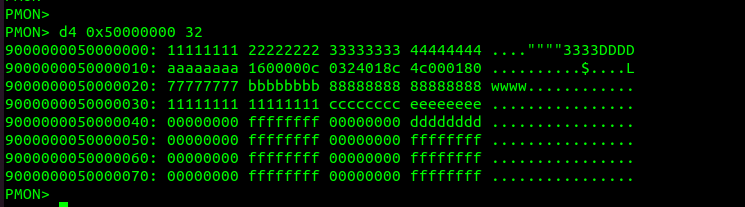
读取设备内的数据：



#### EP端修改内存里的数据值



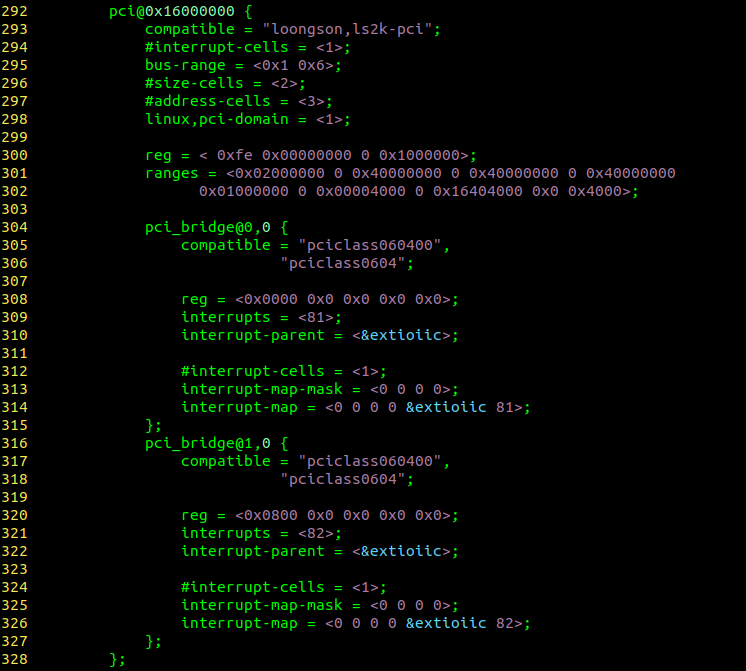
#### RC端再次读取EP设备寄存器值



### PCIE RC模式

#### pmon设备树中PCIE节点

设备树Targets/LS2K500/conf/LS2K500.dts里需打开pcie控制器节点。



## PCI

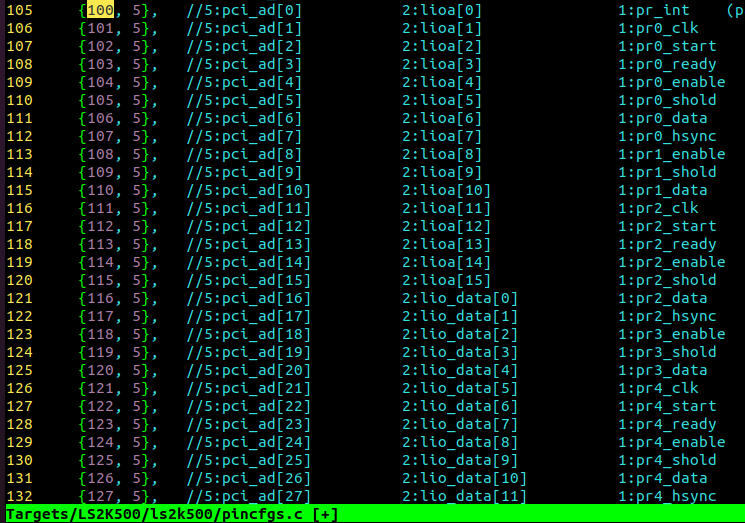
### PCI复用设置

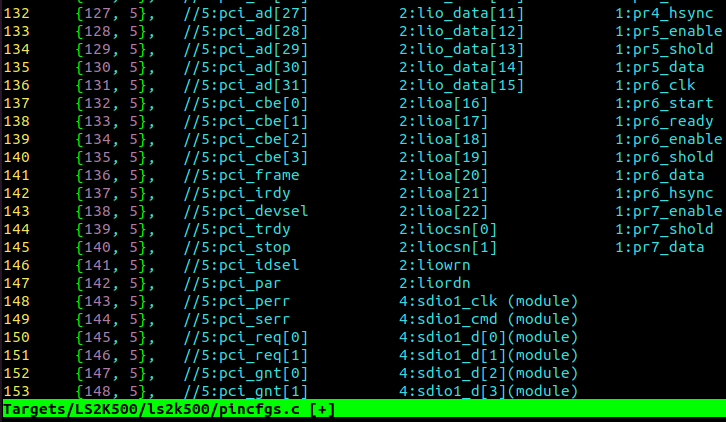
PCI与PRINT、LIO、PWM2存在复用冲突。复用注意事项可查看附录“2K500复用设置注意事项”。PCI复用需要按下述信息进行配置，如果不使用PCI，则需关闭下述PCI选项。

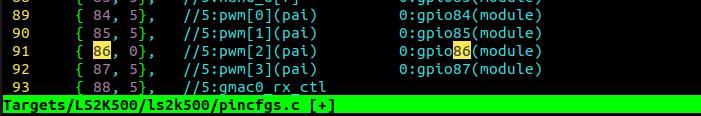
gpio100~148，设置为芯片主功能；

gpio86，设置为gpio功能，用作pci中断引脚。

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需将gpio100~148设置为芯片主功能，gpio86设置为gpio功能。

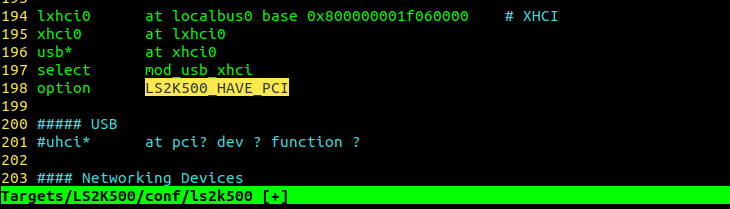






### pmon下PCI配置选项

在配置文件Targets/LS2K500/conf/ls2k500中打开PCI配置项：



### pmon设备树中PCI节点

Targets/LS2K500/conf/LS2K500.dts中打开PCI节点。



## SDIO

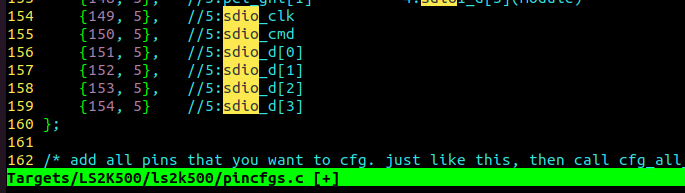
2K500-pai上，使用的SDIO接口为SDIO0。

### SDIO0复用设置

复用注意事项可查看附录“2K500复用设置注意事项”。

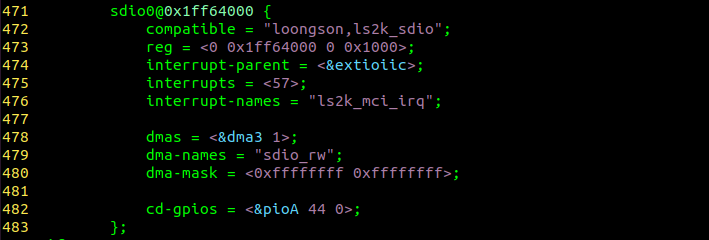
gpio149~154，设置为芯片主功能（SDIO0）。

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需将gpio149~154设置为芯片主功能。



### pmon设备树中SDIO0节点

Targets/LS2K500/conf/LS2K500.dts中打开SDIO0节点。

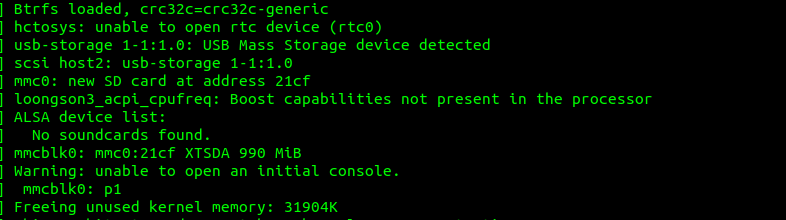


### 内核下SDIO相关配置项

CONFIG\_MMC\_BLOCK、CONFIG\_MMC\_LS2K、

CONFIG\_LS\_APBDMAC、CONFIG\_DMADEVICES。

### 内核启动日志



由上图可见，内核驱动检测到了sdio设备，并在/dev/目录下创建了设备节点mmcblk0，且在设备下存在p1分区，分区设备名为/dev/目录下的mmcblk0p0。

## NAND

NAND在pmon及内核下，设置的ecc校验模式，设置的分区大小情况，需严格保持一致。

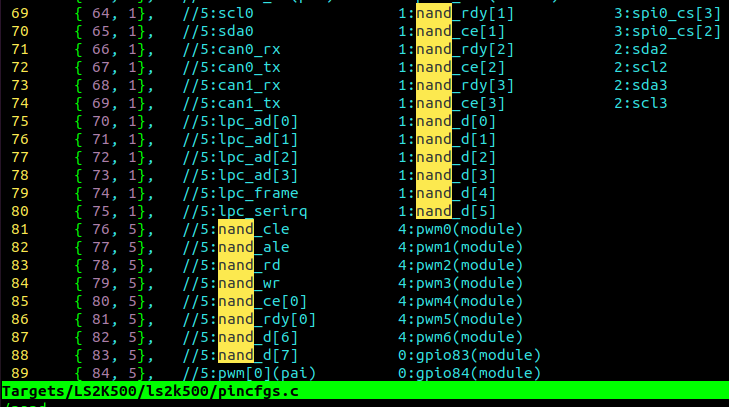
### NAND复用设置

NAND与CAN0、CAN1 、I2C0、I2C2、I2C3 以及 SPI0、SPI1的片选引脚都存在复用冲突。复用注意事项可查看附录“2K500复用设置注意事项”。

gpio64-75，设置成第1复用；

gpio76-83，设置成芯片主功能。

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需将gpio64~75设置为第1复用，将gpio76~83设置为芯片主功能。



### pmon下NAND配置选项

#### nand配置选项

在配置文件Targets/LS2K500/conf/ls2k500中打开下述选项：

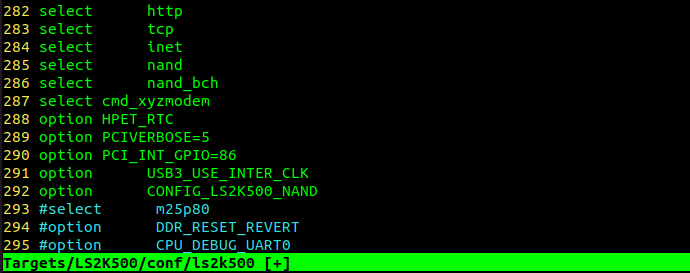
select nand

option CONFIG\_LS2K500\_NAND

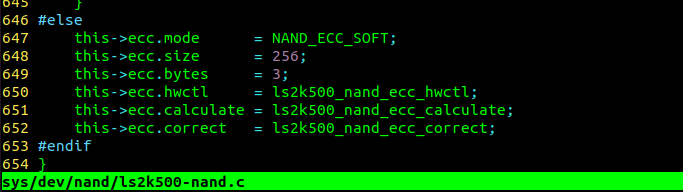
#### nand bch配置选项

在配置文件Targets/LS2K500/conf/ls2k500中打开下述选项：

select nand\_bch

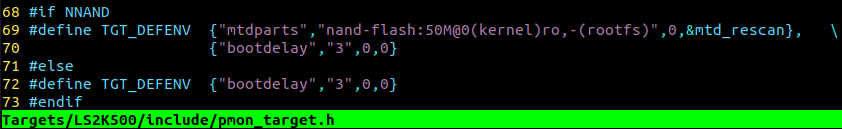


pmon源码下sys/dev/nand/ls2k500-nand.c为NAND的驱动文件，如果在配置文件中关闭了nand\_bch配置，则会在驱动中设置校验模式为ecc软件校验。



#### NAND分区设置

在源码下Targets/LS2K500/include/pmon\_target.h中修改pmon下NAND的默认分区：mtdparts中默认第一个分区为50M，剩下的空间分给第二个分区。可在此修改pmon下的分区设置。

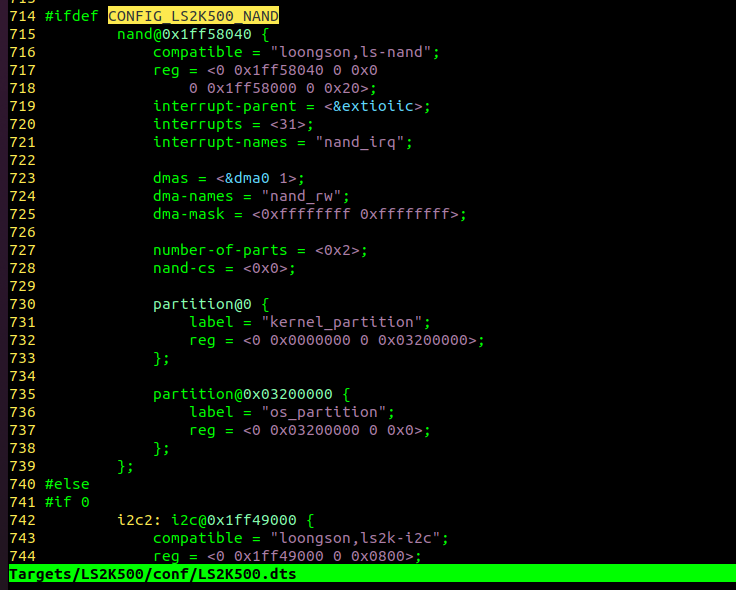


### pmon设备树中NAND节点及分区设置

设备树的nand节点里，也会设置nand分区，但这个是提供给内核驱动进行nand分区设置的，上面则是用于pmon下的nand分区设置。

注：内核下（即设备树中）nand分区设置需与pmon下保持一致。

由下图可知，如果配置文件Targets/LS2K500/conf/ls2k500中打开了了 CONFIG\_LS2K500\_NAND配置项，那么在pmon下的设备树文件Targets/LS2K500/conf/LS2K500.dts里，会将nand节点打开，通过设置设备树里的nand分区信息节点“partition”，来进行内核下的nand分区配置。 如下图，“number-of-parts=<0x2>”,表示设备树一共传递两个分区配置。“partition@0”中的“reg”设置，表示第一个分区偏移为0，分区大小为50M；“partition@0x03200000”中的“reg”设置，表示第二个分区偏移为50M，分区大小为nand剩下的空间。



### 内核下NAND驱动配置

下述配置NAND为bch校验模式：

注：内核下nand校验模式需与pmon下保持一致。

CONFIG\_MTD\_OF\_PARTS、CONFIG\_MTD\_BLOCK、

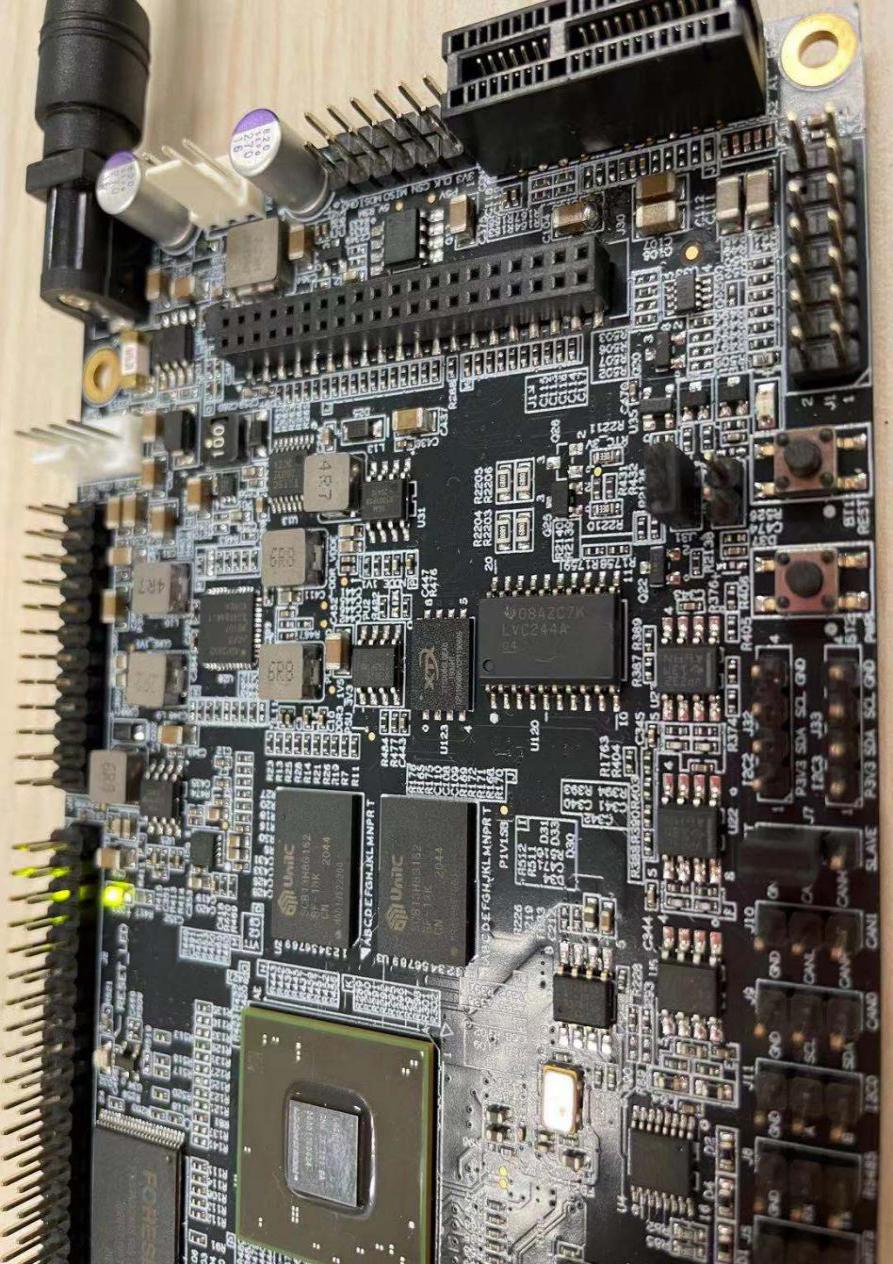
CONFIG\_MTD\_NAND\_LS、CONFIG\_MTD\_NAND\_ECC\_BCH。

### 内核下yaffs2文件系统格式支持

CONFIG\_YAFFS\_FS。

## CAN

2K500-pai上有CAN0与CAN1两个CAN接口。当J31插上跳线帽，则此时板卡设置为CAN功能，可使用CAN0与CAN1；当J31取下跳线帽，此时板卡使用I2C2与I2C3。



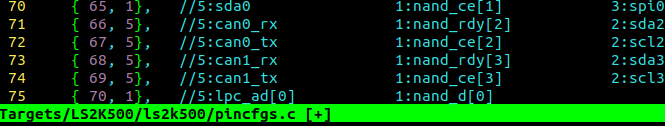
### CAN复用设置

CAN0、CAN1与NAND 、I2C2、I2C3 以及SPI0、SPI1的片选引脚都存在复用冲突。复用注意事项可查看附录“2K500复用设置注意事项”。

gpio66~67，设置为芯片主功能（CAN0）;

gpio68~69，设置为芯片主功能（CAN1）。

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需将gpio66~69设置为芯片主功能。



### pmon设备树中CAN节点

在pmon的配置文件Targets/LS2K500/conf/ls2k500中关闭配置项CONFIG\_2K500\_NAND。



并在设备树Targets/LS2K500/conf/LS2K500.dts里按下图设置“if 0”来打开can0与can1节点。



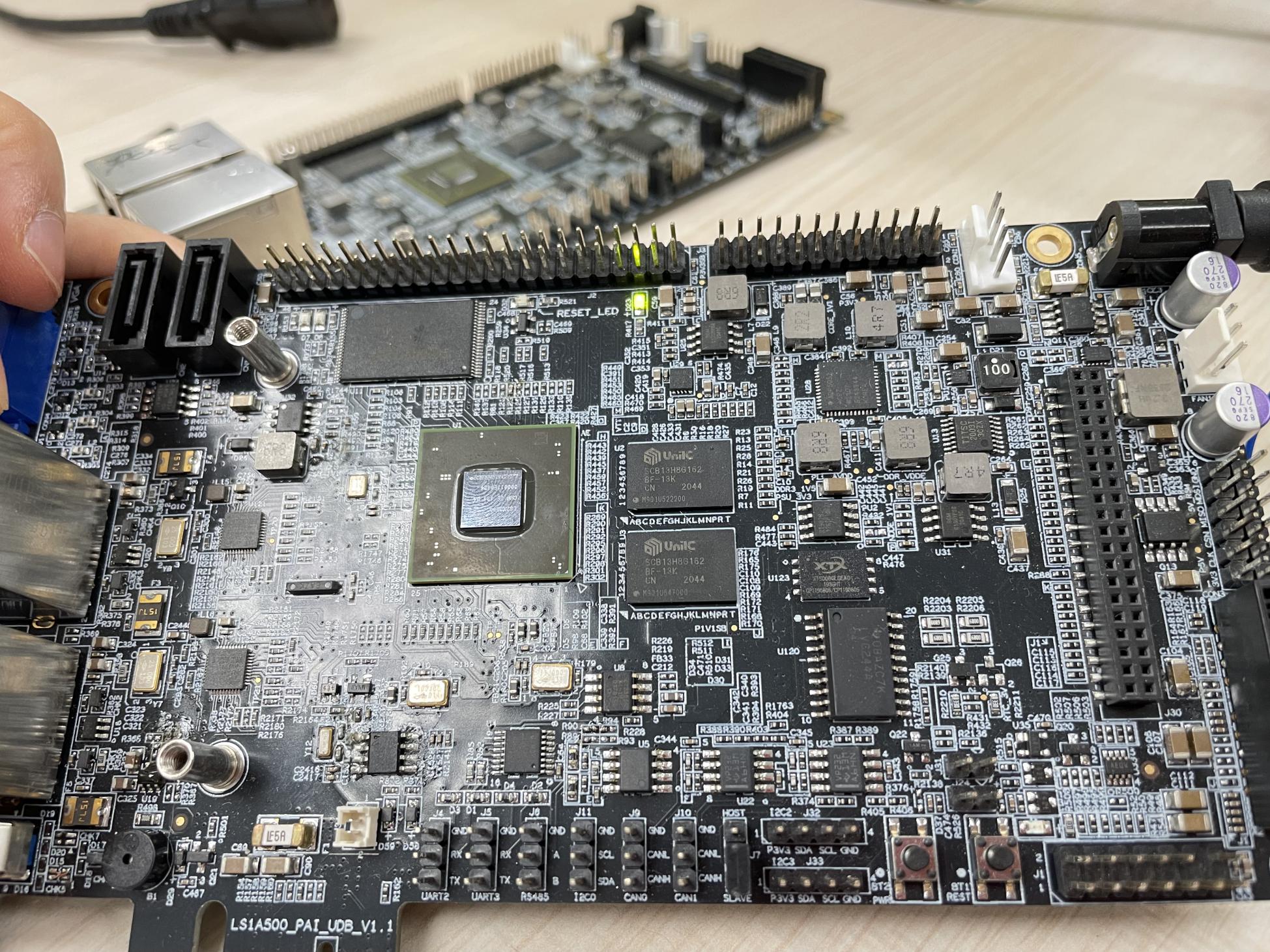
注：can0和can1节点内，status = “disable”;项需按照上图内容注释掉，如果打开，内核将不会初始化这一个设备节点。

### 内核下CAN驱动配置

CONFIG\_CAN\_SJA1000\_PLATFORM。

## I2C

2K500-pai上共有I2C0、I2C2、I2C3、I2C4、I2C5五路i2c。当J31上跳线帽被取下时，此时板卡上I2C2与I2C3可用。其中J32为I2C2，J33为I2C3。



### I2C复用设置

I2C0与NAND、SPI0的片选存在复用冲突；I2C2、I2C3与NAND、CAN0、CAN1以及 SPI0、SPI1的片选引脚都存在复用冲突。复用注意事项可查看附录“2K500复用设置注意事项”。

gpio64-65，设置为芯片主功能（I2C0）；

gpio66-67，设置为第2复用（I2C2）；

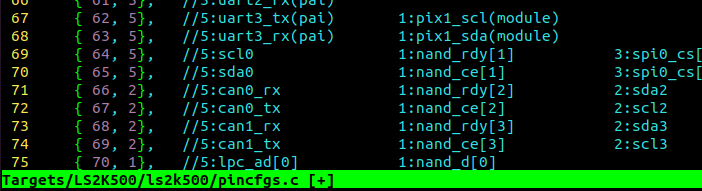
gpio68-69，设置为第2复用（I2C3）；

gpio36-37，设置为第3复用（I2C4(PIX0)）；

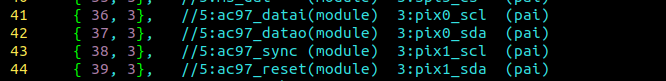
gpio38-39，设置为第3复用（I2C5(PIX1)）。

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需按照下图设置pmon下的复用关系。

I2C0、I2C2、I2C3复用设置如下图：



I2C4、I2C5复用设置如下图：

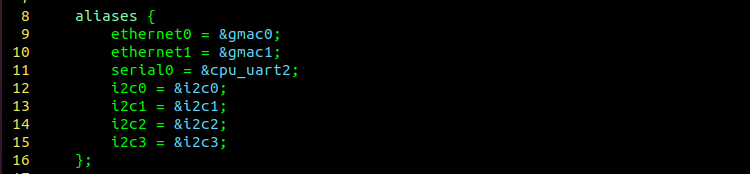


### pmon设备树中I2C节点

在pmon的配置文件Targets/LS2K500/conf/ls2k500中关闭配置项CONFIG\_2K500\_NAND。



在设备树Targets/LS2K500/conf/LS2K500.dts中打开i2c0、i2c2、i2c3设备节点。



如下图，i2c0节点中，status = “disabled”;项被注释掉。i2c1节点中，status = “disabled”;项被打开，则内核会根据设备树，初始化i2c0，不初始化i2c1。



按照上面内容CONFIG\_2K500\_NAND配置项，并且按下图定义”#if 1”，注释掉status参数，则会在设备树中打开i2c2与i2c3节点。

i2c4与i2c5被定义在dc节点内。



### 内核下I2C驱动配置

#### 内核下I2C驱动配置

CONFIG\_I2C\_LS2X。

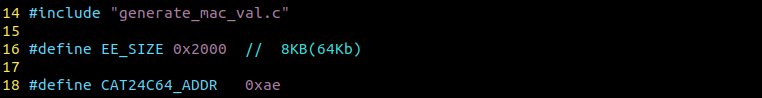
#### 内核下I2C从模式驱动配置

CONFIG\_I2C\_SLAVE、CONFIG\_I2C\_SLAVE\_EEPROM。

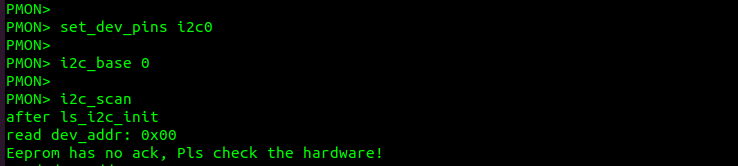
### I2C主/从模式测试用例

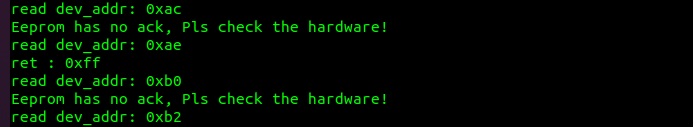
#### pmon下I2C主模式测试用例

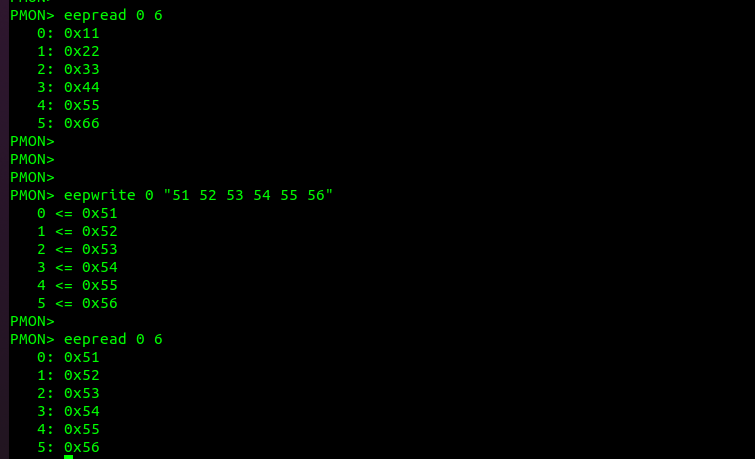
2K500-pai的I2C0的0x57地址挂载了一个64Kb的eeprom设备，地址左移1位，在pmon下被扫到的地址就变为0xae。pmon下的eeprom设备驱动为Targets/LS2K500/dev/eeprom.c，驱动中已将默认设备地址设置为了0xae。



在pmon命令行下，可按照下述命令对eeprom进行操作：







#### 内核下I2C主模式测试用例

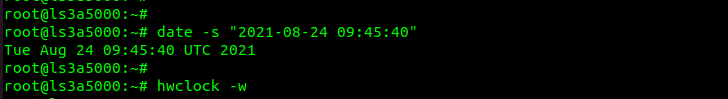
2K500-pai的I2C0上0x68地址接的有一个RTC设备AT8339，兼容DS1338设备驱动。

##### 内核打开RTC驱动配置

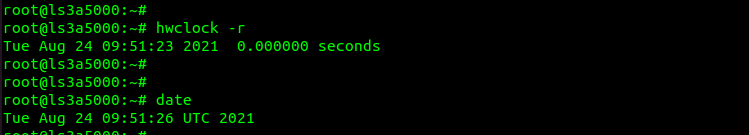
CONFIG\_RTC\_DRV\_DS1307。

##### 设备日志及操作现象

1



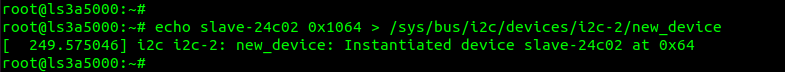
断电6分钟后，上电启动内核读取时间（需连接rtc电池）：



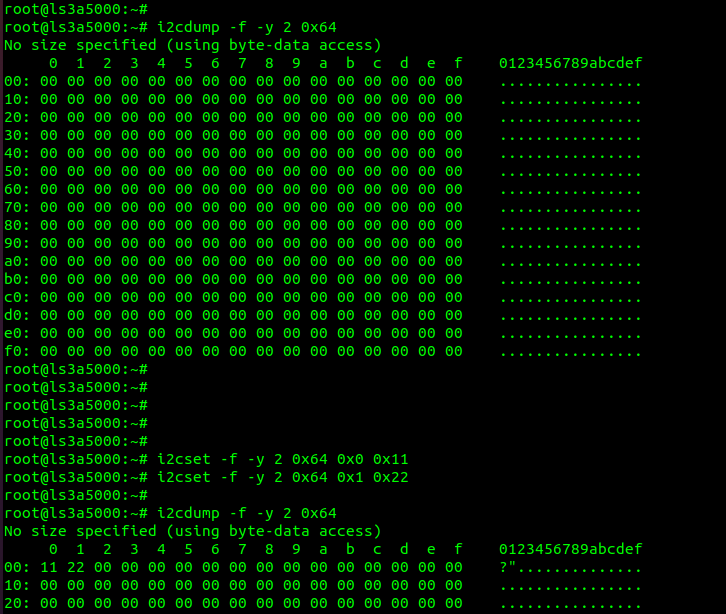
#### 内核下I2C从模式测试用例

根据5.11中第三节的内容，进行I2C从模式的配置后，便可进行下述操作。

将从端I2C2与主端I2C2连接起来，从端设置I2C2为0x64地址。（设置地址时需要或上0x1000，驱动drivers/i2c/i2c-core-base.c的i2c\_sysfs\_new\_device函数处理new\_device设备时，会判断地址flasg是否为I2C\_ADDR\_OFFSET\_SLAVE(0x1000)）



主端对I2C2的0x64设备进行读写操作



注：将同一底板的不同I2C分别设置主、从模式，也可正常通讯；但如果I2C下接的有设备，则该路I2C不可设置为从模式。

## SPI

2K500-pai上有SPI0、SPI1、SPI3三路spi。

### SPI复用设置

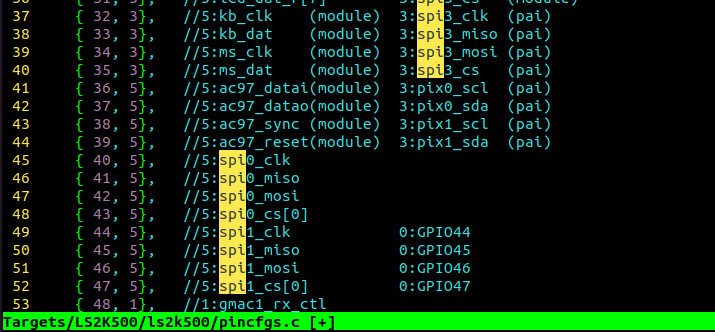
SPI0、SPI1的片选1、片选2、片选3三个引脚信号，分别与NAND、I2C0、I2C2、I2C3、CAN0、CAN1存在复用冲突。复用注意事项可查看附录“2K500复用设置注意事项”。

gpio40~43，设置为芯片主功能，gpio64~66，设置为第3复用；（spi0）

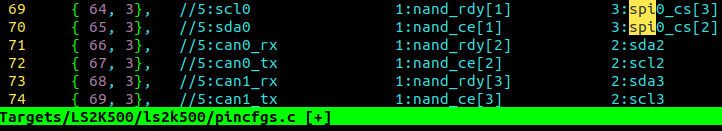
gpio44~47，设置为芯片主功能，gpio67~69，设置为第3复用；（spi1）

gpio32~35，设置为第3复用。（spi3）

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需按照下图设置pmon下的复用关系。



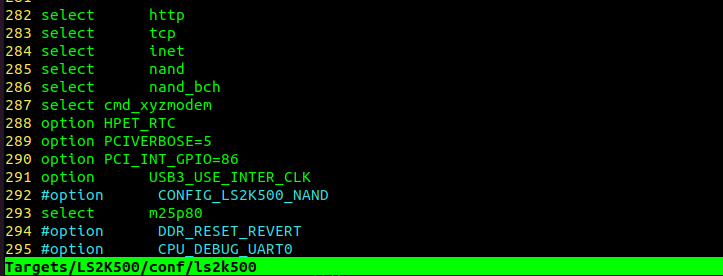
spi0与spi1的片选1/2/3引脚如果没使用到的话，下图中关于spi片选引脚复用的设置可以不用修改。



### pmon下spi norflash配置项

#### norflash配置项

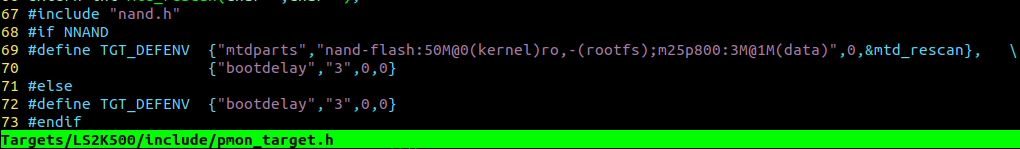
在pmon的配置文件Targets/LS2K500/conf/ls2k500中打开select nand、select m25p80，如果用到了spi0或spi1的cs1/cs2/cs3引脚，则需要按照下图中的配置关闭option CONFIG\_LS2K500\_NAND。



#### norflash分区设置

在Targets/LS2K500/include/pmon\_target.h添加pmon下norflash的默认分区：

SPI0的片选0上的norflash，前1M空间用于存放pmon，如果需要初始化SPI0上的norflash，需要按照图片里的内容设置分区，保留前1M空间。



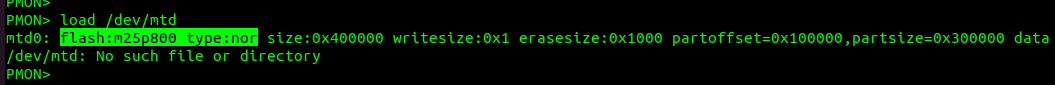
#### norflash操作命令

pmon在Targets/LS2K500/dev/spi\_w.c驱动中，通过ls\_m25p\_probe函数进行设备初始化，在函数里会根据spi\_nand结构体对norflash进行初始化，如下图中，ls2k500\_spi0.base = LS2K500\_SPI0\_BASE，则是初始化spi0上的设备；.chip\_select = 0，表示设备在片选0上。



pmon命令行下，可通过下述命令对设备进行操作：

1. 查看设备分区情况：load /dev/mtd



（可见，此时mtd0为norflash设备的分区，大小为3M）

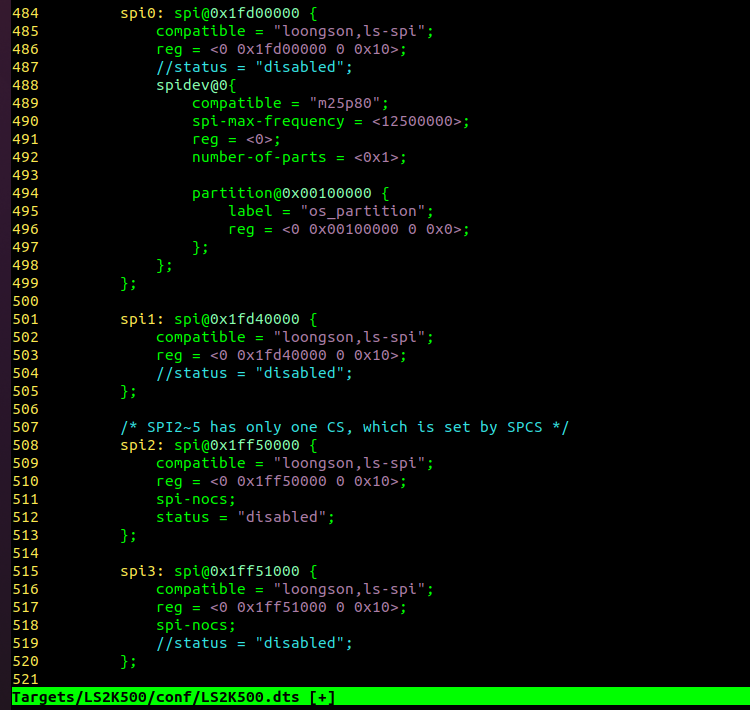
1. 擦除分区：mtd\_erase /dev/mtd0
2. 从u盘中拷贝文件到分区： devcp (usb0,0)/filename /dev/mtd0

### pmon设备树中SPI设备节点

在设备树Targets/LS2K500/conf/LS2K500.dts中打开spi节点。按下图所示，注释掉spi0、spi1、spi3节点中的status = “disabled”项。如果打开，内核会跳过改节点不进行初始化，例如图片中的spi2节点。

如下图所示，spi0下添加了一个norflash设备节点。”reg = <0>;”表示设备在片选0上。”number-of-parts = <0x1>;”表示设备树只传递1个分区信息给内核。“partition@0x00100000”中的“reg”设置，表示第一个分区偏移为1M，分区大小为norflash的剩余空间。

注：内核下norflash分区设置需与pmon下保持一致。



### 内核下SPI及norflash驱动配置

CONFIG\_SPI\_LS、

CONFIG\_MTD\_OF\_PARTS、

CONFIG\_MTD\_BLOCK、

CONFIG\_MTD\_SPI\_NOR、

CONFIG\_MTD\_M25P80。

## RS485/UART3

2K500-pai上，RS485与UART3使用的是同一引脚，板卡上默认使用的功能是RS485，如需使用UART3，需要改硬件。

RS485为半双工工作模式，需要通过gpio45控制数据收发，当gpio45为高电平时，发送数据；当gpio45为低电平时，接收数据。

UART3为全双工工作模式，不需要gpio进行控制。

### RS485 / UART3复用设置

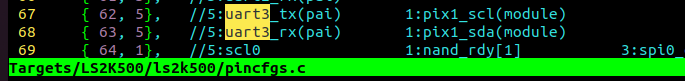
复用注意事项可查看附录“2K500复用设置注意事项”。

gpio62~63，设置为芯片主功能。

gpio45，设置为gpio功能。

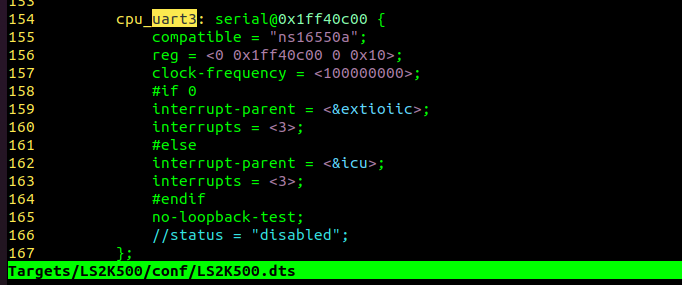
在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需按照下图设置pmon下的复用关系。





### pmon设备树中RS485/UART3节点

在设备树Targets/LS2K500/conf/LS2K500.dts中打开uart3节点。按下图所示，注释掉cpu\_uart3节点中的status = “disabled”项。



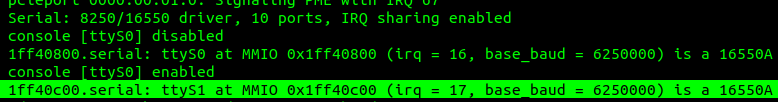
### 内核下串口驱动配置

打开CONFIG\_SERIAL\_8250、

设置CONFIG\_SERIAL\_8250\_NR\_UARTS与 CONFIG\_SERIAL\_8250\_RUNTIME\_UARTS为10

### 现象日志

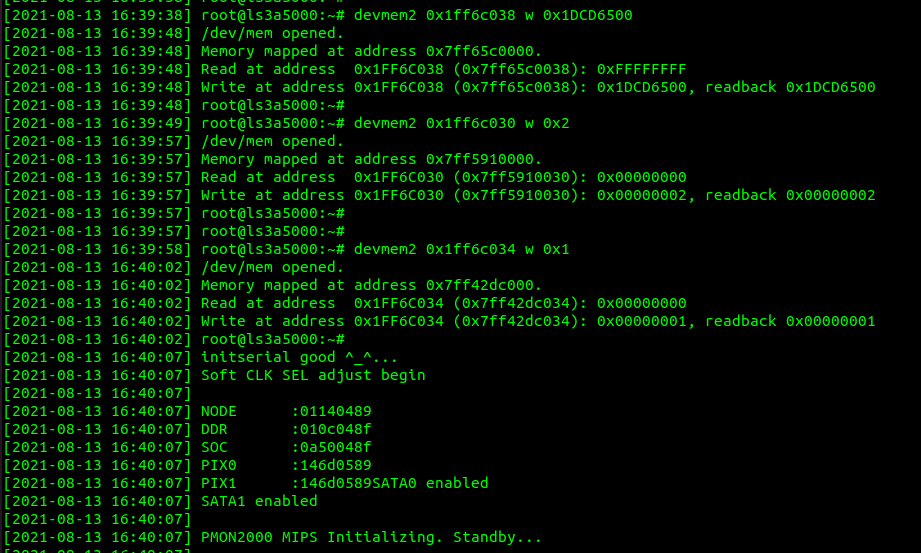
内核下uart3/rs485被初始化为ttyS1节点



## WATCH DOG

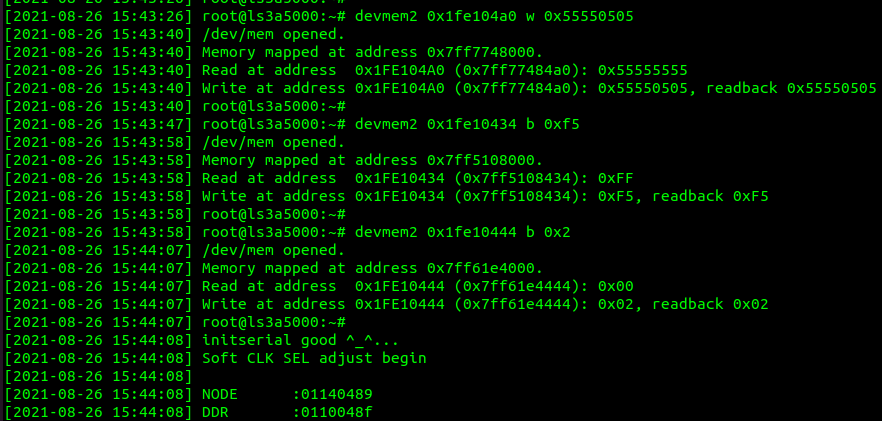
### ACPI内部看门狗（等待5s，系统复位）

写入WD\_Timer寄存器的值为5x100000000=500000000=0x1dcd6500



### 外部看门狗MAX6369（1-3s，系统复位）

外部看门狗通过令gpio33输出高电平，使能看门狗定时器；gpio35进行喂狗操作，保持一种电平，表示不喂狗；不停进行高低电平切换，则表示喂狗操作。



## PWM

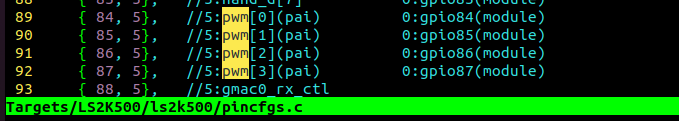
2K500-pai上有PWM0~3四路PWM，其中PWM0设置为了gpio功能，默认做了LCD的背光信号脚，如需使用，需要硬件改电阻；PWM3用作了板卡上电配置，只可做输出功能。

### PWM0~3复用设置

复用注意事项可查看附录“2K500复用设置注意事项”。

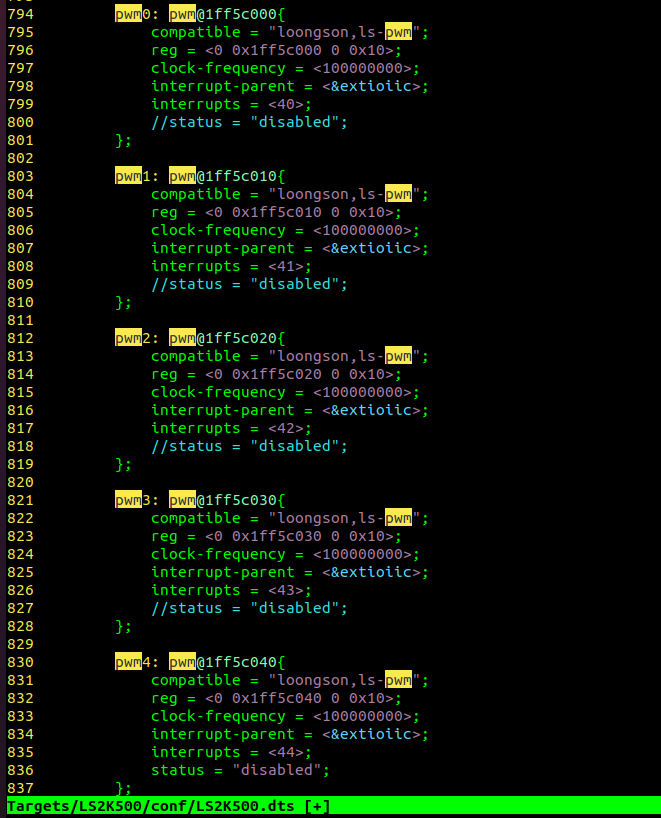
gpio84~87，设置为芯片主功能。（PWM0~3）

在pmon源码下的Targets/LS2K500/ls2k500/pincfgs.c文件中的default\_pin\_cfgs数组里，配置的是板卡pmon默认的gpio复用关系，由上，需按照下图设置pmon下的复用关系。



### pmon设备树中PWM设备节点

在设备树Targets/LS2K500/conf/LS2K500.dts中打开PWM节点。按下图所示，注释掉PWM0~PWM3节点中的status = “disabled”项，内核则会根据节点信息去初始化PWM控制器。如果status项不注释调，内核则不会初始化这些节点，例如PWM4~PWM7。

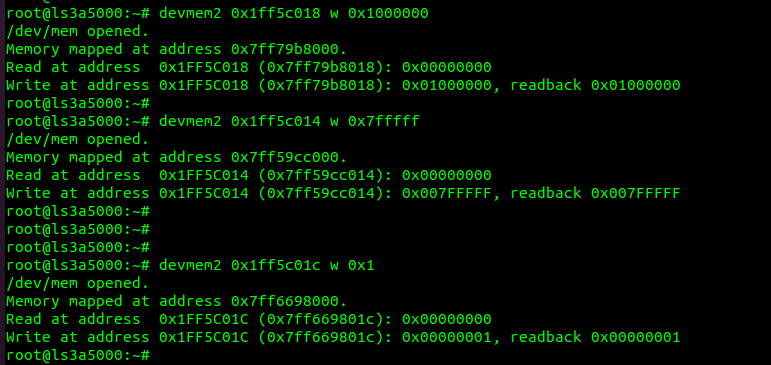


### 内核下PWM驱动

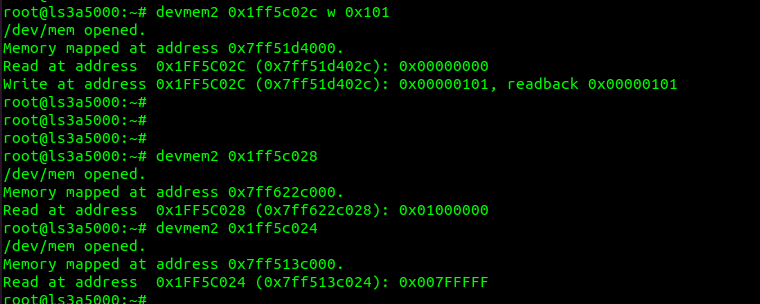
CONFIG\_PWM\_LS。

### PWM输出及捕获配置

#### PWM1输出方波



#### PWM2捕获高低电平



## GPIO中断

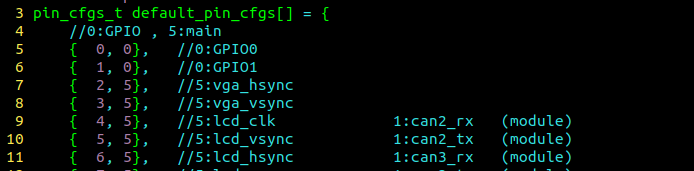
1. 2K500含有155个gpio，其中每4个gpio使用同一个扩展中断，且每组中的第四个gpio不可做gpio中断，即gpio(4 \* N + 3)的gpio号，不可做gpio中断。
2. gpio123~gpio154不可做gpio中断。
3. gpio如果要做中断的话，需要令其外部拉低。
4. 使用相同中断号的一组gpio，只能注册其中一个gpio的中断。

# 附录A： 2K500复用设置注意事项

## 默认复用设置

在pmon源码Targets/LS2K500/ls2k500/pincfgs.c中，default\_pin\_cfgs数组中设置的为引脚的默认复用值。数组中第一个参数为gpio号；第二个参数为引脚复用值，0表示gpio功能，1表示第1复用，2表示第2复用，3表示第3复用，4表示第4复用，5表示芯片主功能。

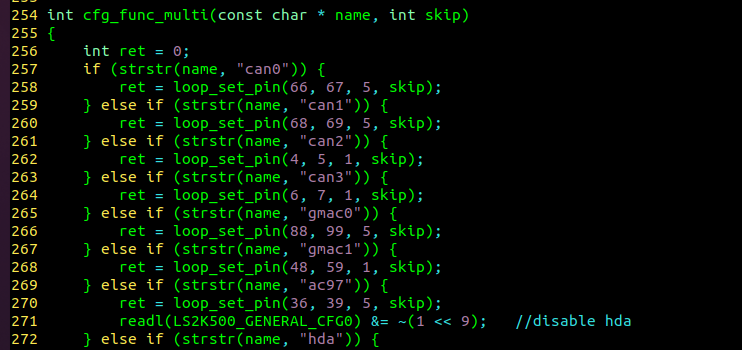
module为嵌入式模块板上，对应的接口功能；pai为2K500-pai上对应的接口功能，如果设计板卡上使用的接口复用不同，则可在数组中对应的进行修改。



## cfg\_func\_multi函数设置

在pmon源码Targets/LS2K500/ls2k500/pincfgs.c中，cfg\_func\_multi函数可以设置接口的功能复用。在pmon下的设备驱动中，会通过该函数在初始化时，对设备复用进行设置，如果设计板卡上使用的接口复用不同，可在函数中对应修改。

例如函数中gmac1，便是令gpio48~59作第1复用功能。



## pmon命令行下调用命令修改复用配置

板卡调试阶段，如果不想多次烧写pmon去修改接口复用，则可在pmon命令行下通过set\_dev\_pins命令，对设备复用进行设置。命令会调用(2)中的cfg\_func\_multi函数，通过函数中设置的引脚复用值去设置接口复用。

例： set\_dev\_pins i2c0

## 复用冲突处理

注：接口的复用引脚只能设置1组，其余同组功能的引脚复用需关闭。

例如I2C0，gpio48~49的第3复用，以及gpio64~65的芯片主功能，这些都可设置为I2C0功能，但这些gpio引脚，同时必须只能有一组作为I2C0功能。

即以I2C0为例，如果gpio64~65做了芯片主功能，则gpio48~49便不可设置为第3复用，否则将存在接口功能冲突问题，导致i2c0无法被正常使用。