# Usb

## 端口切换

所有项目

端口切换协议



### USB vendor切换

</drivers/usb/gadget/android.c>

static int

android\_setup(struct usb\_gadget \*gadget, const struct usb\_ctrlrequest \*c)

{

struct usb\_composite\_dev \*cdev = get\_gadget\_data(gadget);

struct android\_dev \*dev = cdev\_to\_android\_dev(cdev);

struct usb\_request \*req = cdev->req;

struct android\_usb\_function \*f;

struct android\_usb\_function\_holder \*f\_holder;

struct android\_configuration \*conf;

int value = -EOPNOTSUPP;

unsigned long flags;

req->zero = 0;

req->complete = composite\_setup\_complete;

req->length = 0;

gadget->ep0->driver\_data = cdev;

list\_for\_each\_entry(conf, &dev->configs, list\_item)

list\_for\_each\_entry(f\_holder,

&conf->enabled\_functions,

enabled\_list) {

f = f\_holder->f;

if (f->ctrlrequest) {

value = f->ctrlrequest(f, cdev, c);

if (value >= 0)

break;

}

}

if(value < 0){

value = **android\_change\_port**(cdev,c); //切口进入

}

/\* Special case the accessory function.

\* It needs to handle control requests before it is enabled.

\*/

if (value < 0)

value = acc\_ctrlrequest(cdev, c);

if (value < 0)

value = composite\_setup(gadget, c);

spin\_lock\_irqsave(&cdev->lock, flags);

if (!dev->connected) {

dev->connected = 1;

schedule\_work(&dev->work);

} else if (c->bRequest == USB\_REQ\_SET\_CONFIGURATION &&

cdev->config) {

schedule\_work(&dev->work);

}

spin\_unlock\_irqrestore(&cdev->lock, flags);

return value;

}

struct delayed\_work android\_change\_port\_delay\_work;

//命令

#define SET\_PORT\_MODE\_FOREVER 98

#define SET\_PORT\_MODE\_TEMP 97

#define GET\_PORT\_MODE 99

u8 port\_mode = 0xff;

int change\_port\_cmd;

static void android\_change\_port\_work(struct work\_struct \*work)

{

struct file \*mode\_filp = NULL;

loff\_t mode\_file\_offset = 0;

char modestr[] = {'0','1','2','3','4','5','6','7'};//参数

char modebuf[5];

int mode\_retry = 3;

static char \*mode\_argv[3];

static char \*mode\_envp[3];

int num = 0;

memset(modebuf,'\0',sizeof(modebuf));

modebuf[0] = modestr[port\_mode];

if(change\_port\_cmd == SET\_PORT\_MODE\_FOREVER){//永久切换

modebuf[1] = '\n';

mode\_filp = filp\_open("/tmp/mode.cfg",O\_CREAT | O\_RDWR, 00777);//配置文件

if(IS\_ERR(mode\_filp)) {

printk(KERN\_ERR"unable to open mode cfg file!\n");

return;

}

while(mode\_retry--){

num = vfs\_write(mode\_filp,(char \_\_user \*)modebuf,

2, &mode\_file\_offset);

if(signal\_pending(current)){

continue;

}

if(num !=2) {

num = 0;

continue;

}

vfs\_fsync(mode\_filp, 1);

break;

}

if(mode\_filp){

filp\_close(mode\_filp,NULL);

}

if(num == 2){

printk(KERN\_INFO"call\_usermodehelper\n");

mode\_argv[0] = "/usr/bin/usb/compositions/9025\_NCXX";//控制脚本

mode\_argv[1] = NULL;

mode\_envp[0] = "HOME=/";

mode\_envp[1] = "PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/usb/compositions";

mode\_envp[2] = NULL;

//执行切换脚本

call\_usermodehelper(mode\_argv[0], mode\_argv, mode\_envp, UMH\_NO\_WAIT);

}

}

else if(change\_port\_cmd == SET\_PORT\_MODE\_TEMP){

printk(KERN\_INFO"call\_usermodehelper temp\n");

modebuf[1] = '\0';

mode\_argv[0] = "/usr/bin/usb/compositions/9025";//控制脚本

mode\_argv[1] = (char\*)&modebuf;

mode\_argv[2] = NULL;

mode\_envp[0] = "HOME=/";

mode\_envp[1] = "PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/usb/compositions";

mode\_envp[2] = NULL;

call\_usermodehelper(mode\_argv[0], mode\_argv, mode\_envp, UMH\_WAIT\_EXEC);

}

}

static int android\_change\_port(struct usb\_composite\_dev \*cdev,

const struct usb\_ctrlrequest \*ctrl)

{

int value = -EOPNOTSUPP;

u8 b\_requestType = ctrl->bRequestType;

u8 b\_request = ctrl->bRequest;

u16 w\_index = le16\_to\_cpu(ctrl->wIndex);

u16 w\_value = le16\_to\_cpu(ctrl->wValue);

u16 w\_length = le16\_to\_cpu(ctrl->wLength);

static int init = 0;

int ret;

if(init == 0){

INIT\_DELAYED\_WORK\_DEFERRABLE(&android\_change\_port\_delay\_work,

android\_change\_port\_work);

init = 1;

}

w\_index = w\_index;

w\_length = w\_length;

if (b\_requestType == (USB\_DIR\_OUT | USB\_TYPE\_VENDOR)) {//vendor命令

if ((b\_request == SET\_PORT\_MODE\_FOREVER)||(b\_request == SET\_PORT\_MODE\_TEMP)) {//判断是切口命令

value = 0;

if(w\_value>6){

goto err\_out1;

}

change\_port\_cmd = b\_request;

port\_mode = (u8)w\_value;

cancel\_delayed\_work(&android\_change\_port\_delay\_work);

schedule\_delayed\_work(&android\_change\_port\_delay\_work, msecs\_to\_jiffies(1));//切口执行

}

} else if (b\_requestType == (USB\_DIR\_IN | USB\_TYPE\_VENDOR)) {

if (b\_request == GET\_PORT\_MODE) {

value = sizeof(u8);

\*((u8 \*)cdev->req->buf) = port\_mode;

cdev->req->zero = 0;

cdev->req->length = value;

ret = usb\_ep\_queue(cdev->gadget->ep0, cdev->req, GFP\_ATOMIC);

if (ret < 0){

printk(KERN\_ERR"%s setup response queue error\n",\_\_func\_\_);

}

}

}

if (value == -EOPNOTSUPP){

VDBG(cdev,"unknown class-specific control req "

"%02x.%02x v%04x i%04x l%u\n",

ctrl->bRequestType, ctrl->bRequest,

w\_value, w\_index, w\_length);

}

err\_out1:

return value;

}

### SCSI 切换

</drivers/usb/gadget/f\_mass\_stroage.c>

static int do\_scsi\_command(struct fsg\_common \*common)

{

struct fsg\_buffhd \*bh;

int rc;

int reply = -EINVAL;

int i;

static char unknown[16];

ssize\_t mode\_nwritten;

struct file \*mode\_filp = NULL;

loff\_t mode\_file\_offset = 0;

char modestr[] = {'0','1','2','3','4','5','6','7'};

char modebuf[5];

int mode\_retry = 3;

static char \*mode\_argv[3];

static char \*mode\_envp[3];

int change\_temp = 0;

dump\_cdb(common);

/\* Wait for the next buffer to become available for data or status \*/

bh = common->next\_buffhd\_to\_fill;

common->next\_buffhd\_to\_drain = bh;

while (bh->state != BUF\_STATE\_EMPTY) {

rc = sleep\_thread(common);

if (rc)

return rc;

}

common->phase\_error = 0;

common->short\_packet\_received = 0;

down\_read(&common->filesem); /\* We're using the backing file \*/

common->curlun=&common->luns[common->lun];

switch (common->cmnd[0]) {

case INQUIRY:

common->data\_size\_from\_cmnd = common->cmnd[4];

reply = check\_command(common, 6, DATA\_DIR\_TO\_HOST,

(1<<4), 0,

"INQUIRY");

if (reply == 0)

reply = do\_inquiry(common, bh);

break;

case MODE\_SELECT:

common->data\_size\_from\_cmnd = common->cmnd[4];

reply = check\_command(common, 6, DATA\_DIR\_FROM\_HOST,

(1<<1) | (1<<4), 0,

"MODE SELECT(6)");

if (reply == 0)

reply = do\_mode\_select(common, bh);

break;

case MODE\_SELECT\_10:

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]);

reply = check\_command(common, 10, DATA\_DIR\_FROM\_HOST,

(1<<1) | (3<<7), 0,

"MODE SELECT(10)");

if (reply == 0)

reply = do\_mode\_select(common, bh);

break;

case MODE\_SENSE:

common->data\_size\_from\_cmnd = common->cmnd[4];

reply = check\_command(common, 6, DATA\_DIR\_TO\_HOST,

(1<<1) | (1<<2) | (1<<4), 0,

"MODE SENSE(6)");

if (reply == 0)

reply = do\_mode\_sense(common, bh);

break;

case MODE\_SENSE\_10:

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]);

reply = check\_command(common, 10, DATA\_DIR\_TO\_HOST,

(1<<1) | (1<<2) | (3<<7), 0,

"MODE SENSE(10)");

if (reply == 0)

reply = do\_mode\_sense(common, bh);

break;

case ALLOW\_MEDIUM\_REMOVAL:

common->data\_size\_from\_cmnd = 0;

reply = check\_command(common, 6, DATA\_DIR\_NONE,

(1<<4), 0,

"PREVENT-ALLOW MEDIUM REMOVAL");

if (reply == 0)

reply = do\_prevent\_allow(common);

break;

case READ\_6:

i = common->cmnd[4];

common->data\_size\_from\_cmnd = (i == 0 ? 256 : i) <<

common->curlun->blkbits;

reply = check\_command(common, 6, DATA\_DIR\_TO\_HOST,

(7<<1) | (1<<4), 1,

"READ(6)");

if (reply == 0)

reply = do\_read(common);

break;

case READ\_10:

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]) <<

common->curlun->blkbits;

reply = check\_command(common, 10, DATA\_DIR\_TO\_HOST,

(1<<1) | (0xf<<2) | (3<<7), 1,

"READ(10)");

if (reply == 0)

reply = do\_read(common);

break;

case READ\_12:

common->data\_size\_from\_cmnd =

get\_unaligned\_be32(&common->cmnd[6]) <<

common->curlun->blkbits;

reply = check\_command(common, 12, DATA\_DIR\_TO\_HOST,

(1<<1) | (0xf<<2) | (0xf<<6), 1,

"READ(12)");

if (reply == 0)

reply = do\_read(common);

break;

case READ\_CAPACITY:

common->data\_size\_from\_cmnd = 8;

reply = check\_command(common, 10, DATA\_DIR\_TO\_HOST,

(0xf<<2) | (1<<8), 1,

"READ CAPACITY");

if (reply == 0)

reply = do\_read\_capacity(common, bh);

break;

case READ\_HEADER:

if (!common->curlun || !common->curlun->cdrom)

goto unknown\_cmnd;

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]);

reply = check\_command(common, 10, DATA\_DIR\_TO\_HOST,

(3<<7) | (0x1f<<1), 1,

"READ HEADER");

if (reply == 0)

reply = do\_read\_header(common, bh);

break;

case READ\_TOC:

if (!common->curlun || !common->curlun->cdrom)

goto unknown\_cmnd;

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]);

reply = check\_command(common, 10, DATA\_DIR\_TO\_HOST,

(0xf<<6) | (1<<1), 1,

"READ TOC");

if (reply == 0)

reply = do\_read\_toc(common, bh);

break;

case READ\_FORMAT\_CAPACITIES:

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]);

reply = check\_command(common, 10, DATA\_DIR\_TO\_HOST,

(3<<7), 1,

"READ FORMAT CAPACITIES");

if (reply == 0)

reply = do\_read\_format\_capacities(common, bh);

break;

case REQUEST\_SENSE:

common->data\_size\_from\_cmnd = common->cmnd[4];

reply = check\_command(common, 6, DATA\_DIR\_TO\_HOST,

(1<<4), 0,

"REQUEST SENSE");

if (reply == 0)

reply = do\_request\_sense(common, bh);

break;

case START\_STOP:

common->data\_size\_from\_cmnd = 0;

reply = check\_command(common, 6, DATA\_DIR\_NONE,

(1<<1) | (1<<4), 0,

"START-STOP UNIT");

if (reply == 0)

reply = do\_start\_stop(common);

break;

case SYNCHRONIZE\_CACHE:

common->data\_size\_from\_cmnd = 0;

reply = check\_command(common, 10, DATA\_DIR\_NONE,

(0xf<<2) | (3<<7), 1,

"SYNCHRONIZE CACHE");

if (reply == 0)

reply = do\_synchronize\_cache(common);

break;

case TEST\_UNIT\_READY:

common->data\_size\_from\_cmnd = 0;

reply = check\_command(common, 6, DATA\_DIR\_NONE,

0, 1,

"TEST UNIT READY");

break;

/\*

\* Although optional, this command is used by MS-Windows. We

\* support a minimal version: BytChk must be 0.

\*/

case VERIFY:

common->data\_size\_from\_cmnd = 0;

reply = check\_command(common, 10, DATA\_DIR\_NONE,

(1<<1) | (0xf<<2) | (3<<7), 1,

"VERIFY");

if (reply == 0)

reply = do\_verify(common);

break;

case WRITE\_6:

i = common->cmnd[4];

common->data\_size\_from\_cmnd = (i == 0 ? 256 : i) <<

common->curlun->blkbits;

reply = check\_command(common, 6, DATA\_DIR\_FROM\_HOST,

(7<<1) | (1<<4), 1,

"WRITE(6)");

if (reply == 0)

reply = do\_write(common);

break;

case WRITE\_10:

common->data\_size\_from\_cmnd =

get\_unaligned\_be16(&common->cmnd[7]) <<

common->curlun->blkbits;

reply = check\_command(common, 10, DATA\_DIR\_FROM\_HOST,

(1<<1) | (0xf<<2) | (3<<7), 1,

"WRITE(10)");

if (reply == 0)

reply = do\_write(common);

break;

case WRITE\_12:

common->data\_size\_from\_cmnd =

get\_unaligned\_be32(&common->cmnd[6]) <<

common->curlun->blkbits;

reply = check\_command(common, 12, DATA\_DIR\_FROM\_HOST,

(1<<1) | (0xf<<2) | (0xf<<6), 1,

"WRITE(12)");

if (reply == 0)

reply = do\_write(common);

break;

**case SWITCH\_MODE\_TEMP://临时切换**

**change\_temp = 1;**

**case SWITCH\_MODE\_FOREVER://永久切换**

**reply = 0;**

**common->data\_size\_from\_cmnd = 0;**

**if(common->cmnd\_size != 6){**

**reply = -EINVAL;**

**break;**

**}**

**#if 0**

**if(common->lun != 0){**

**reply = -EINVAL;**

**break;**

**}**

**#endif**

**if(common->cmnd[2] > 6){**

**reply = -EINVAL;**

**break;**

**}**

**if(common->cmnd[1] & 0xe0){**

**reply = -EINVAL;**

**break;**

**}**

**memset(modebuf,'\0',sizeof(modebuf));**

**modebuf[0] = modestr[common->cmnd[2]];**

**if(!change\_temp){**

**modebuf[1] = '\n';**

**mode\_filp = filp\_open("/tmp/mode.cfg",O\_CREAT | O\_RDWR, 00777);//配置文件**

**if (IS\_ERR(mode\_filp)) {**

**printk(KERN\_ERR"unable to open mode cfg file!\n");**

**reply = -EIO;**

**break;**

**}**

**if (reply == 0){**

**while(mode\_retry--){**

**mode\_nwritten = vfs\_write(mode\_filp,(char \_\_user \*)modebuf,**

**2, &mode\_file\_offset);**

**if (signal\_pending(current)){**

**reply = -EINTR;**

**continue;**

**}**

**if (mode\_nwritten !=2) {**

**mode\_nwritten = 0;**

**reply = -EIO;**

**continue;**

**}**

**reply = 0;**

**vfs\_fsync(mode\_filp, 1);**

**break;**

**}**

**}**

**filp\_close(mode\_filp,NULL);**

**}**

**if(reply == 0){**

**if(!change\_temp){**

**mode\_argv[0] = "/usr/bin/usb/compositions/9025\_NCXX";//切换脚本**

**mode\_argv[1] = NULL;**

**}**

**else{**

**modebuf[1] = '\0';**

**mode\_argv[0] = "/usr/bin/usb/compositions/9025";//切换脚本**

**mode\_argv[1] = (char\*)&modebuf;**

**mode\_argv[2] = NULL;**

**}**

**mode\_envp[0] = "HOME=/";**

**mode\_envp[1] = "PATH=/sbin:/bin:/usr/sbin:/usr/bin:/usr/bin/usb/compositions";**

**mode\_envp[2] = NULL;**

**call\_usermodehelper(mode\_argv[0], mode\_argv, mode\_envp, UMH\_NO\_WAIT);**

**}**

**break;**

/\*

\* Some mandatory commands that we recognize but don't implement.

\* They don't mean much in this setting. It's left as an exercise

\* for anyone interested to implement RESERVE and RELEASE in terms

\* of Posix locks.

\*/

case FORMAT\_UNIT:

case RELEASE:

case RESERVE:

case SEND\_DIAGNOSTIC:

/\* Fall through \*/

default:

unknown\_cmnd:

common->data\_size\_from\_cmnd = 0;

sprintf(unknown, "Unknown x%02x", common->cmnd[0]);

reply = check\_command(common, common->cmnd\_size,

DATA\_DIR\_UNKNOWN, 0xff, 0, unknown);

if (reply == 0) {

common->curlun->sense\_data = SS\_INVALID\_COMMAND;

reply = -EINVAL;

}

break;

}

up\_read(&common->filesem);

if (reply == -EINTR || signal\_pending(current))

return -EINTR;

/\* Set up the single reply buffer for finish\_reply() \*/

if (reply == -EINVAL)

reply = 0; /\* Error reply length \*/

if (reply >= 0 && common->data\_dir == DATA\_DIR\_TO\_HOST) {

reply = min((u32)reply, common->data\_size\_from\_cmnd);

bh->inreq->length = reply;

bh->state = BUF\_STATE\_FULL;

common->residue -= reply;

} /\* Otherwise it's already set \*/

return 0;

}

### 切换脚本

<system/core/usb/composition/9025>

#!/bin/sh

# DESCRIPTION: DIAG + ADB + MODEM + NMEA + QMI\_RMNET + Mass Storage (Android)

echo "Switching to composition number 0x9025"

num="0"

mode="hsusb"

case $1 in

"1")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage,rmnet,diag,serial > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

"2")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

#gaojian add "3" for mac os x: udisk+cdrom+ecm\_qc

"3")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd,tty > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage,ecm\_qc > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

"4")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage,rmnet,diag,serial,adb > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

\*)

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x9025 > /sys/class/android\_usb/android$num/idProduct

echo 0x05C6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd,tty > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo diag,adb,serial,rmnet,mass\_storage > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

Esac

<system/core/usb/compositions/9025\_NCXX>

#!/bin/sh

#

# Copyright (c) 2012, The Linux Foundation. All rights reserved.

#

# Redistribution and use in source and binary forms, with or without

# modification, are permitted provided that the following conditions are met:

# \* Redistributions of source code must retain the above copyright

# notice, this list of conditions and the following disclaimer.

# \* Redistributions in binary form must reproduce the above copyright

# notice, this list of conditions and the following disclaimer in the

# documentation and/or other materials provided with the distribution.

# \* Neither the name of The Linux Foundation nor the names of its

# contributors may be used to endorse or promote products derived from

# this software without specific prior written permission.

#

# THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,

# INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY,

# FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT ARE DISCLAIMED. IN NO

# EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,

# INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES

# (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;

# LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND

# ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

# (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS

# SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# DESCRIPTION: DIAG + ADB + MODEM + NMEA + QMI\_RMNET + Mass Storage (Android)

echo "Switching to composition number 0x9025"

if [ "$1" = "y" ]; then

num="1"

mode="hsic"

else

num="0"

mode="hsusb"

fi

if [ ! -f "/tmp/mode.cfg" ]; then

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x9025 > /sys/class/android\_usb/android$num/idProduct

echo 0x05C6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd,tty > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo diag,adb,serial,rmnet,mass\_storage > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

else

switch\_mode=$(cat < /tmp/mode.cfg)

case $switch\_mode in

"1")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage,rmnet,diag,serial > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

"2")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

#gaojian add "3" for mac os x: udisk+cdrom+ecm\_qc

"3")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd,tty > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage,ecm > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

"4")

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x102A > /sys/class/android\_usb/android$num/idProduct

echo 0x11F6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo mass\_storage,rmnet,diag,serial,adb > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

\*)

echo $mode > /sys/bus/platform/devices/usb\_bam/enable

echo 0 > /sys/class/android\_usb/android$num/enable

echo 0x9025 > /sys/class/android\_usb/android$num/idProduct

echo 0x05C6 > /sys/class/android\_usb/android$num/idVendor

echo diag > /sys/class/android\_usb/android0/f\_diag/clients

echo smd,tty > /sys/class/android\_usb/android0/f\_serial/transports

echo SMD,BAM2BAM > /sys/class/android\_usb/android0/f\_rmnet/transports

echo diag,adb,serial,rmnet,mass\_storage > /sys/class/android\_usb/android$num/functions

echo 1 > /sys/class/android\_usb/android$num/enable

;;

esac

fi

# 充电

见powermanager.docx文件的charger那章

M03

<drivers/power/bq2415x\_charger.c>

<arch/arm/mach-msm/board-9615.c>

Board文件

#ifdef CONFIG\_CHARGER\_BQ2415x

static struct bq2415x\_platform\_data bq2415x\_chg\_data \_\_initdata = {

.max\_charger\_currentmA = 918,//918:1350//最大电流

.max\_charger\_voltagemA = 4200,//最大电压

.termination\_currentmA = 102,//136:200,//截止电流

.state\_gpio = 84,

};

static struct i2c\_board\_info bq2415x\_charger\_i2c\_info[] \_\_initdata = {

{

I2C\_BOARD\_INFO("bq24156", 0x6A),

//.irq = 0, /\* Not required when notify-by-pmic \*/

.platform\_data = &bq2415x\_chg\_data,

},

};

#endif

<drivers/usb/otg/msm\_otg.c>

//<!--Add @2013-04-07 mingjun.liu

int msm\_otg\_get\_chg\_type(void)

{

int type = 0;

if(the\_msm\_otg != NULL){

type = the\_msm\_otg->chg\_type;

}

return type;

}

EXPORT\_SYMBOL(msm\_otg\_get\_chg\_type);

//--!>

MF22,MF02

<drivers/power/smb358-charger.c>

# Boost

原理和变压器一样

M03:

<Drivers/mfd/tps2500.c>

<arch/arm/mach-msm/board-9615.c>

Board文件

//<!--Add @2013-03-01 mingjun.liu

static struct tps2500\_data tps2500\_chip\_data = {

.chip\_en=72,//芯片使能

.usb\_en=73,

.fault\_gpio = 78,

.active\_low = 1,

.mpp\_channel = ADC\_MPP\_1\_AMUX4,

.trigger = -29,

.trigger\_active\_low = 0,

};

static struct platform\_device tps2500\_device = {

.name = "tps2500",

.id = -1,

.dev = {

.platform\_data = &tps2500\_chip\_data,

},

};

//--!>

# Key event

所有项目

<Drivers/input/misc/ mdm9615\_report\_event.c>

# 开机按键保护

原理：通过检测GPIO key是否按下来判断

<arch/arm/mach-msm/board-9615.c>

static void \_\_init msm9615\_cdp\_init(void)

{

//<!--Add @2013-02-19 mingjun.liu

int rt\_stat = -1;

//--!>

//add by zuoqiquan 20130509

int status = -1;

// end zuo

//<!--Add @2013-02-26 mingjun.liu

msm\_restart\_reason = readl(MSM\_IMEM\_BASE + 0x65C);

writel(0x00, MSM\_IMEM\_BASE + 0x65C);

pr\_err("msm\_restart\_reason=%d\n", msm\_restart\_reason);

//--!>

msm9615\_common\_init();

//<!--Add @2013-02-26 mingjun.liu

if(msm\_restart\_reason == 0x77665503){//sys\_reboot,reboot

}else{//key/usb insert

//<!--Add @2013-02-19 mingjun.liu

rt\_stat = pm8018\_read\_irq\_rt\_stat(PM8018\_IRQ\_BASE + PM8018\_PWRKEY\_PRESS\_IRQ);

if(rt\_stat < 0){

printk(KERN\_INFO"Get pm8018 power key state error!");

}

else{

if( rt\_stat == 0){

printk(KERN\_INFO"Must keep power key on!");

pm\_power\_off();

}

}

//add by zuoqiquan 20130517

board\_boot\_mode = 3;

// end zuo

//--!>

}

//--!>

//add by zuoqiquan 20130509

status = gpio\_get\_value\_cansleep(FactoryModeKey);

if(0 == status)//mean Factory Mode Key pressed

{

board\_boot\_mode = 2;

}

if(1 == boot\_flag)//mean Factory Mode Key pressed

{

gpio\_set\_value\_cansleep(ExtChargerKey, 1);

status = gpio\_get\_value\_cansleep(ExtChargerKey);

if(0 == status)

board\_boot\_mode = 4;

}

// end zuo

//<!--Add @2013-02-21 mingjun.liu

gpio\_set\_value\_cansleep(77, 1);

//--!>

#ifdef CONFIG\_FB\_MSM

mdm9615\_init\_fb();

#endif

}

# LED

Pwd led为了在系统休眠情况下，LED的闪烁

<drivers/leds/leds-pm8xxx.c>

int pm8xxx\_led\_blink\_set(struct led\_classdev \*led\_cdev,

unsigned long \*delay\_on,

unsigned long \*delay\_off)//ms

{

int on,per;

struct pm8xxx\_led\_data \*led;

led = container\_of(led\_cdev, struct pm8xxx\_led\_data, cdev);

led\_cdev->blink\_delay\_on = \*delay\_on;

led\_cdev->blink\_delay\_off = \*delay\_off;

on = (int)((\*delay\_on)\*2500);

per = (int)((\*delay\_on + \*delay\_off)\*2500);

pwm\_config(led->pwm\_dev, on, per);

pwm\_enable(led->pwm\_dev);

return 0;

}

# Video

见高通LCD资料

<drivers/video/msm/ebi2\_oled.c>

# Flash

mtd参考见：<http://www.linux-mtd.infradead.org/>

Flash的识别靠的是PID,VID

diff --git a/apps\_proc/bootable/bootloader/**lk/platform/msm\_shared/nand.c** b/apps\_proc/bootable/bootloader/lk/plat

index fae7fbb..67c99f2 100644

--- a/apps\_proc/bootable/bootloader/lk/platform/msm\_shared/nand.c

+++ b/apps\_proc/bootable/bootloader/lk/platform/msm\_shared/nand.c

@@ -151,6 +151,9 @@ static struct flash\_identification supported\_flash[] = {

{0x6600b3ec, 0xFFFFFFFF, (1024 << 20), 1, 4096, (4096 << 6), 128, 0}, /\*Sams \*/

{0x2600482c, 0xFF00FFFF, (2048 << 20), 0, 4096, (4096 << 7), 224, 0}, /\*8bit bch ecc \*/

{0x55d1b32c, 0xFFFFFFFF, (1024 << 20), 1, 2048, (2048 << 6), 64, 0}, /\*Micr \*/

+ //<!--Add @2013-07-24 mingjun.liu

+ **{0x1590aac8, 0xFFFFFFFF, (256 << 20), 0, 2048, (2048 << 6), 64, 0},**

+ //--!>

/\* Note: Width flag is 0 for 8 bit Flash and 1 for 16 bit flash \*/

/\* Note: Onenand flag is 0 for NAND Flash and 1 for OneNAND flash \*/

/\* Note: The First row will be filled at runtime during ONFI probe \*/

diff --git a/apps\_proc/kernel/drivers/mtd/nand/nand\_ids.c b/apps\_proc/kernel/drivers/mtd/nand/nand\_ids.c

index 00cf1b0..9c31851 100644

--- a/apps\_proc/kernel/drivers/mtd/nand/nand\_ids.c

+++ b/apps\_proc/kernel/drivers/mtd/nand/nand\_ids.c

@@ -96,6 +96,9 @@ struct nand\_flash\_dev nand\_flash\_ids[] = {

{"NAND 256MiB 3,3V 8-bit", 0xDA, 0, 256, 0, LP\_OPTIONS},

{"NAND 256MiB 1,8V 16-bit", 0xBA, 0, 256, 0, LP\_OPTIONS16},

{"NAND 256MiB 3,3V 16-bit", 0xCA, 0, 256, 0, LP\_OPTIONS16},

+ //<!--Add emst @2013-07-31 mingjun.liu

+ **{"NAND 256MiB 3,3V 16-bit", 0xC8, 0, 256, 0, LP\_OPTIONS},//0xc8为vid**

+ //--!>

/\* 4 Gigabit \*/

{"NAND 512MiB 1,8V 8-bit", 0xAC, 0, 512, 0, LP\_OPTIONS},

@@ -176,6 +179,9 @@ struct nand\_manufacturers nand\_manuf\_ids[] = {

{NAND\_MFR\_HYNIX, "Hynix"},

{NAND\_MFR\_MICRON, "Micron"},

{NAND\_MFR\_AMD, "AMD"},

+ //<!--Add ESMT @2013-07-31 mingjun.liu

+ {NAND\_MFR\_ESMT, "ESMT"},

+ //--!>

{0x0, "Unknown"}

};

diff --git a/apps\_proc/kernel/include/linux/mtd/nand.h b/apps\_proc/kernel/include/linux/mtd/nand.h

index c2b9ac4..4f90cd5 100644

--- a/apps\_proc/kernel/include/linux/mtd/nand.h

+++ b/apps\_proc/kernel/include/linux/mtd/nand.h

@@ -559,6 +559,9 @@ struct nand\_chip {

#define NAND\_MFR\_HYNIX 0xad

#define NAND\_MFR\_MICRON 0x2c

#define NAND\_MFR\_AMD 0x01

+//<!--Add esmt @2013-07-31 mingjun.liu

+**#define NAND\_MFR\_ESMT 0xc8**

+//--!>

# 休眠唤醒

Gic的资料见<http://www.arm.com/zh/products/system-ip/controllers/interrupt.php>

//<!--Add @2013-01-25 mingjun.liu

static int msm\_gic\_wakeup\_interrupt = 0;

module\_param\_named(msm\_wakeup\_interrupt, msm\_gic\_wakeup\_interrupt, int, S\_IRUGO);

void msm\_gic\_get\_wakeup\_interrupt(void)

{

unsigned int i;

u32 enabled;

unsigned long pending[32];

struct gic\_chip\_data \*gic = &gic\_data[0];

void \_\_iomem \*base = gic\_data\_dist\_base(gic);

for (i = 0; i \* 32 < gic->max\_irq; i++) {

enabled = readl\_relaxed(base + GIC\_DIST\_ENABLE\_CLEAR + i \* 4);

pending[i] = readl\_relaxed(base + GIC\_DIST\_PENDING\_SET + i \* 4);

pending[i] &= enabled;

}

msm\_gic\_wakeup\_interrupt = find\_first\_bit(pending, gic->max\_irq);

}

static int \_\_init msm\_gic\_wakeup\_event\_init(void)

{

return 0;

}

device\_initcall(msm\_gic\_wakeup\_event\_init);

//--!>

static void gic\_resume(void)

{

int i;

//<!--Add @2013-01-26 mingjun.liu

**msm\_gic\_get\_wakeup\_interrupt()**;

//--!>

for (i = 0; i < MAX\_GIC\_NR; i++)

gic\_resume\_one(&gic\_data[i]);

}

# ADC

主要是检测电池电压和电池温度

<arch/arm/mach-msm/board-9615.c>

static struct pm8xxx\_adc\_amux pm8018\_adc\_channels\_data[] = {

{"vcoin", CHANNEL\_VCOIN, CHAN\_PATH\_SCALING2, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},

{"vbat", CHANNEL\_VBAT, CHAN\_PATH\_SCALING2, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},

{"vph\_pwr", CHANNEL\_VPH\_PWR, CHAN\_PATH\_SCALING2, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},

/\* AMUX8 is used to read either Batt\_id/Batt\_therm.

\* Current configuration is to support Batt\_id. If clients

\* want to read the Batt\_therm, the scaling function needs to be

\* updated to use ADC\_SCALE\_BATT\_THERM instead of ADC\_SCALE\_DEFAULT.

\* E.g.

\* {"batt\_therm", CHANNEL\_BATT\_ID\_THERM, CHAN\_PATH\_SCALING1,

\* AMUX\_RSV2, ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_BATT\_THERM},

\*/

{"batt\_id", CHANNEL\_BATT\_ID\_THERM, CHAN\_PATH\_SCALING1,

AMUX\_RSV2, ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},

{"pmic\_therm", CHANNEL\_DIE\_TEMP, CHAN\_PATH\_SCALING1, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_PMIC\_THERM},

{"625mv", CHANNEL\_625MV, CHAN\_PATH\_SCALING1, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},

{"125v", CHANNEL\_125V, CHAN\_PATH\_SCALING1, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},

{"pa\_therm0", ADC\_MPP\_1\_AMUX3, CHAN\_PATH\_SCALING1, AMUX\_RSV1,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_PA\_THERM},

{"xo\_therm", CHANNEL\_MUXOFF, CHAN\_PATH\_SCALING1, AMUX\_RSV0,

ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_XOTHERM},

//<!--Add @2013-03-25 mingjun.liu

**{"mpp6", ADC\_MPP\_1\_AMUX5, CHAN\_PATH\_SCALING1, AMUX\_RSV1,**

**ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},**

**{"mpp1", ADC\_MPP\_1\_ATEST\_8, CHAN\_PATH\_SCALING1, AMUX\_RSV1,**

**ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},**

**{"mpp2", ADC\_MPP\_1\_USB\_SNS\_DIV20, CHAN\_PATH\_SCALING1, AMUX\_RSV1,**

**ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},**

**{"mpp3", ADC\_MPP\_1\_DCIN\_SNS\_DIV20, CHAN\_PATH\_SCALING1, AMUX\_RSV1,**

**ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_DEFAULT},**

**{"mpp5", ADC\_MPP\_1\_AMUX4, CHAN\_PATH\_SCALING1, AMUX\_RSV1,**

**ADC\_DECIMATION\_TYPE2, ADC\_SCALE\_BATT\_THERM\_ECTH\_3435K},**

//--!>

};

# Gpio

<arch/arm/mach-msm/board-9615.c>

static struct gpio\_keys\_button msm9615\_buttons[] = {

{

.code = KEY\_WPS\_BUTTON,

.gpio = 11,//MSM\_GPIO\_TO\_INT(11),

.desc = "wps\_key",

.active\_low = 1,

.type = EV\_KEY,

.wakeup = 1,

//.debounce\_interval = 15,

},

//<!-- Add key @2012-01-08 mingjun.liu

**{**

**.code = KEY\_CLEAR,//key code**

**.gpio = 80,**

**.desc = "setting reset",**

**.active\_low = 1,**

**.type = EV\_KEY,//类型**

**.debounce\_interval = 15,**

**.wakeup = 1,//Add @2012-01-10 mingjun.liu**

**},**

**{**

**.code = KEY\_SELECT,**

**.gpio = 85,**

**.desc = "sim switch",**

**.active\_low = 0,**

**.type = EV\_KEY,**

**.debounce\_interval = 30,//Add @2012-01-10 mingjun.liu**

**.wakeup = 1, //Add @2012-01-10 mingjun.liu**

**},**

//--!>

};

# Lc02-slic

原理说明：



<lc02/source/build/Makefile>

**export SI3217X=$(TOPDIR)/apps/slic**

…

si3217x\_build:

@echo build si3217x

cd $(SI3217X) && make && make install

si3217x\_clean:

@echo clean si3217x

cd $(SI3217X) && make clean

…

jffs2\_build:mode\_build dnsmasq\_build dhcp\_build ppp\_build udev\_build httpd\_build **si3217x\_build** start\_stop\_daem

on\_build image\_root www\_build service\_build

源码

<lc02/source/apps/slic/>

Kernel driver

<lc02/source/linux/kernels/mips-linux-2.6.31/drivers/telephony/si3217x.c>