Memory Card Boot Solution for A10

V1.0

2012.05.15

Revision History

Version	Date	Section/ Page	Description
v1.0	2012.5.15		Initial version

Table of Contents

1. Overview

With the aid of some firmware flash tools, such as Livesuit/PhoenixUsbPro, and PhoenixCard, etc, firmware can be mounted to the device memory via USB or memory cards.

Firmware can be flashed to an emmc-interfaced memory cards in Startup Mode, so that the cards can be used to boot a system via card controllers. Firmware can be flashed to memory cards in Product Mode as well, so that the cards can be used repeatedly to mount the firmware to device memory, say NAND or eMMC.
What's notable is that on the A10 platform, the pluggable SD card can start a device through card0 controller and card2 controller, while the non-pluggable card such as emmc can only start a device via card2 controller.

This documentation illuminates on how to add memory card boot method to current A10 solutions with three parts: the first and foremost part details modifications required to be made to current A10 solutions, the EVB-MMC Solution and the NAND Solution, the second part briefs how to make a firmware, and the last part is about the firmware flashing.

2. Modifications to A10 Solutions

2.1. Modifications Based on the EVB-MMC Solution

2.1.1. Add a New Solution

Following modifications can be made to the *device/softwinner/crane-evb_mmc* in SDK so that the A10 EVB can boot from a memory card.

1) Copy *crane-evb_mmc* directory to a same-level directory, and rename it as *xxxx_mmc*;
2) Revise the solution name-involved characters in following six files in the newly generated directory: *<solution>.mk, vendorsetup, AndroidProducts.mk, package.sh, BoardConfig.mk, recovery/Android.mk*, etc. Also of note is the use of dash "-" and underscore "_".
3) implement following command :

```
source    build/envsetup.sh
lunch
```

And you will see the new solution in the list.

2.1.2. Modify the Packing

Rename *lichee/tools/pack/chips/sun4i/configs/crane/evb_mmc/* as *lichee/tools/pack/chips/sun4i/configs/crane/xxxx_mmc/*.

This directory includes three files: sysconfig.fex stores information about the firmware partition and Livesuit configuration parameters (Reference can be made to the script notes); sysconfig1.fex is used to configure some pin and module-relevant parameters; and env.cfg stores some uboot-related configuration parameters. Detailed modifications are illustrated below.

2.1.2.1. sysconfig.fex

Storage_type is a vital parameter that defines whether firmware is booted from MMC or NAND when it's mounted in Livesuit. By default, or when storage_type=0, the firmware is booted from NAND, and when storage_type=1, the firmware is booted from MMC.

2.1.2.2. sysconfig1.fex

1） Card0 Boot

```
[card_boot0_para]
card_ctrl          = 0
card_high_speed    = 1
card_line          = 4
sdc_d1             = port:PF0<2><1><default><default>
sdc_d0             = port:PF1<2><1><default><default>
sdc_clk            = port:PF2<2><1><default><default>
sdc_cmd            = port:PF3<2><1><default><default>
sdc_d3             = port:PF4<2><1><default><default>
sdc_d2             = port:PF5<2><1><default><default>
```

2） Card2 Boot

```
[card_boot2_para]
card_ctrl          = 2
card_high_speed    = 1
card_line          = 4
sdc_cmd            = port:PC6<3><1>
sdc_clk            = port:PC7<3><1>
sdc_d0             = port:PC8<3><1>
sdc_d1             = port:PC9<3><1>
sdc_d2             = port:PC10<3><1>
sdc_d3             = port:PC11<3><1>
```

2.1.2.3. env.cfg

Parameters stored in env.cfg include:

1） Bootdelay: indicates how long will uboot wait before it enters command line boot mode (only need in debug);

2） Bootcmd: the default boot command alternate to uboot command line boot. Generally no modification is required;

3） Setargs: a parameter of bootcmd, includes console, nand_root, mmc_root, init, loglevel, etc;

4） Boot_normal: a parameter of bootcmd, defines the location and size of the loaded kernel mirror, and the location to boot kernel;

5） Boot_recovery & boot_fastboot: not used when in normal condition;

2.2. Modifications Based on the NAND Solution

2.2.1. Add a New Solution

1） Find the original NAND solution directory, copy it to the same directory and rename it.
For example, copy *crane-evb* directory to the same directory and rename it as *crane-evb_mmc*;

2） Revise the solution name-involved characters in following six files in the newly generated directory: *<solution>.mk, vendorsetup, AndroidProducts.mk, package.sh, BoardConfig.mk, recovery/Android.mk*, etc. Also of note is the use of dash "-" and underscore "_".
For example, the *crane-evb* and *crane_evb* in *crane_evb.mk*, *vendorsetup*, *AndroidProducts.mk*, *BoardConfig.mk*, *recovery/Android.mk* and *package.sh* all should be altered to *crane-evb_mmc* and *crane_evb_mmc* respectively.

3） implement following command in android directory

```
source    build/envsetup.sh
lunch
```

and you will see the new solution in list.

2.2.2. Modify the New Solution

To make the new solution bootable from a memory card, following modifications should be made:

1） Modify *init.sun4i.rc* and *ueventd.sun4i.rc*, override the NAND partition with MMC partition based on following mapping list:

nanda —— mmcblk0p2
nandb —— mmcblk0p5
nandc —— mmcblk0p6
nandd —— mmcblk0p7
nande —— mmcblk0p8
nandf —— mmcblk0p9
…and so on.

2） Remove or command out following code:

```
format_userdata /dev/block/nand........
```

3） Modify *vold.fstab* based on following instructions:

Original:

```
dev_mount     sdcard     /mnt/sdcard     auto /devices/virtual/block/nandi
dev_mount     extsd      /mnt/extsd      auto /devices/platform/sunxi-mmc.1/mmc_host
    /devices/platform/sunxi-mmc.0/mmc_host
```

Modified:

```
dev_mount    sdcard    /mnt/sdcard    auto /devices/platform/sunxi-mmc.0/mmc_host
dev_mount    extsd     /mnt/extsd     auto /devices/platform/sunxi-mmc.1/mmc_host
```

The number behind sunxi-mmc should be fixed accordingly. If boot from card2, then it should be modified to:

```
dev_mount    sdcard    /mnt/sdcard    auto /devices/platform/sunxi-mmc.2/mmc_host
dev_mount    extsd     /mnt/extsd     auto /devices/platform/sunxi-mmc.1/mmc_host
/devices/platform/sunxi-mmc.0/mmc_host
```

If the card controllers are insufficient, please consult technology personnel for the modification. False modification may render the load failure of external card or connection failure between PC and devices.

If the firmware generated is required to be alterable by the firmware modification tool we provide, a file named *preinstall.sh* should be created (or overridden if there was one) with following contents inside:

```
#!/system/bin/busybox sh

echo "do preinstall job"
BUSYBOX="/system/bin/busybox"
BOOT_MEDIA="nanda"
if [ ! -e /data/system.notfirstrun ]; then
        /system/bin/sh /system/bin/pm preinstall /system/preinstall
        # copy android modify tool files
        mkdir /mnt/nanda
        if [    -e /dev/block/mmcblk0p2 ]; then
                BOOT_MEDIA="mmcblk0p2"
        fi
        mount -t vfat /dev/block/$BOOT_MEDIA    /mnt/nanda
#       $BUSYBOX cp /mnt/nanda/vendor/initlogo.rle /
        $BUSYBOX cp /mnt/nanda/vendor/system/build.prop /system/
        $BUSYBOX cp /mnt/nanda/vendor/system/media/bootanimation.zip /system/media/
        $BUSYBOX cp /mnt/nanda/vendor/system/usr/keylayout/*.kl /system/usr/keylayout/
        sync
        umount /mnt/nanda
        rmdir /mnt/nanda
        $BUSYBOX touch /data/system.notfirstrun
fi
echo "preinstall ok"
```

2.2.3. Modify the Packing

The packing modification is identical with that of EVB-MMC solution. See here for Details.

3. Firmware Compilation and Packing

The firmware compilation and packing is similar to that of NAND solution.
In lichee, type in command:

```
$./build.sh –p sun4i_crane –k 3.0
```

After compilation, go into the android directory:

```
$source build/ envsetup.sh
lunch
```

and select the right solution number.

Then execute:

```
make-all
```

After that, execute：

```
pack
```

And the firmware will end up in lichee/tools/pack/.

4. Firmware Flashing

This section will introduce some firmware flashing tools, such as *Livesuit, PhoenixUsbPro,* and *PhoenixCard,* and their detailed usage.

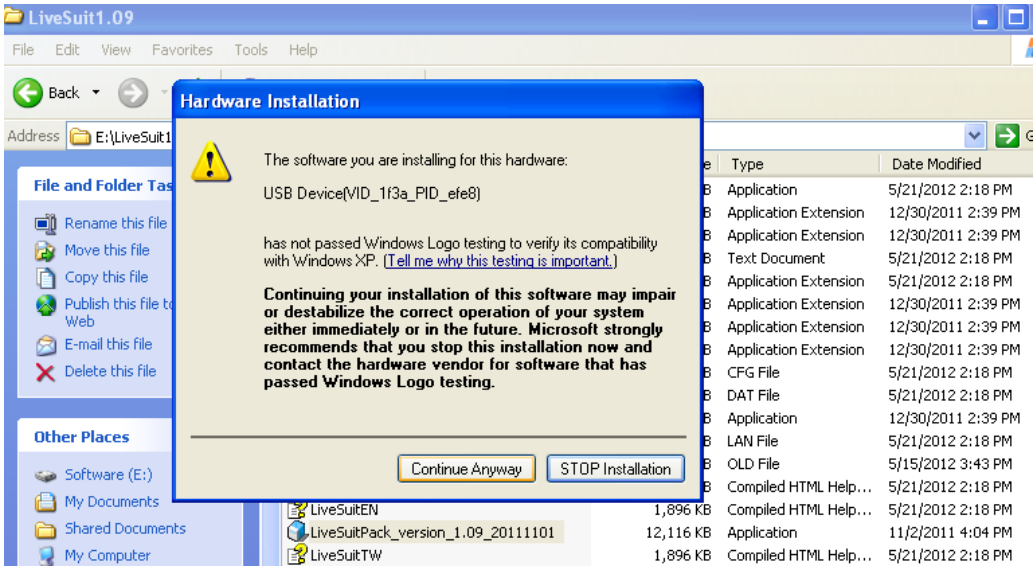
4.1. USB-Based Firmware Flashing

Livesuit or PhoenixUsbPro can be used to flash firmware to a device via the USB OTG on PCs.
Note that Livesuit and PhoenixUsbPro are not supposed to be used at the same time.

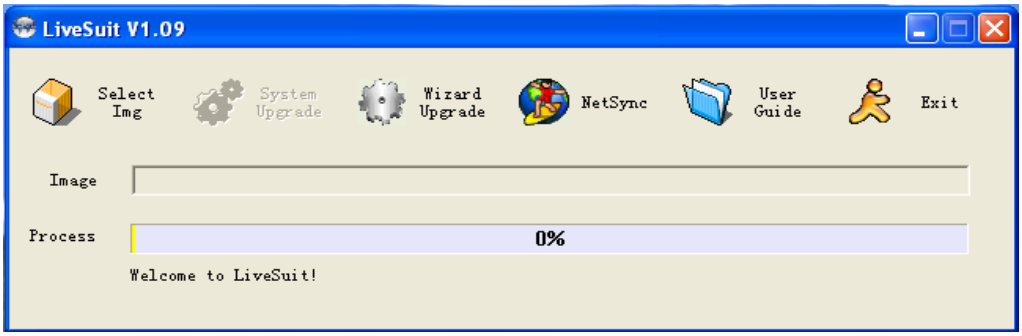
Here takes Livesuit1.07 on Windows platform as an example to exemplify the flashing procedure:

1） Acquire the LivesuitPack.exe from the solution provider , and then locate it in a folder separately;

2） Double click it, then Livesuit software will be automatically installed in this very directory. During the installation, you will be asked whether to continue the install or not, select “Continue Anyway”;

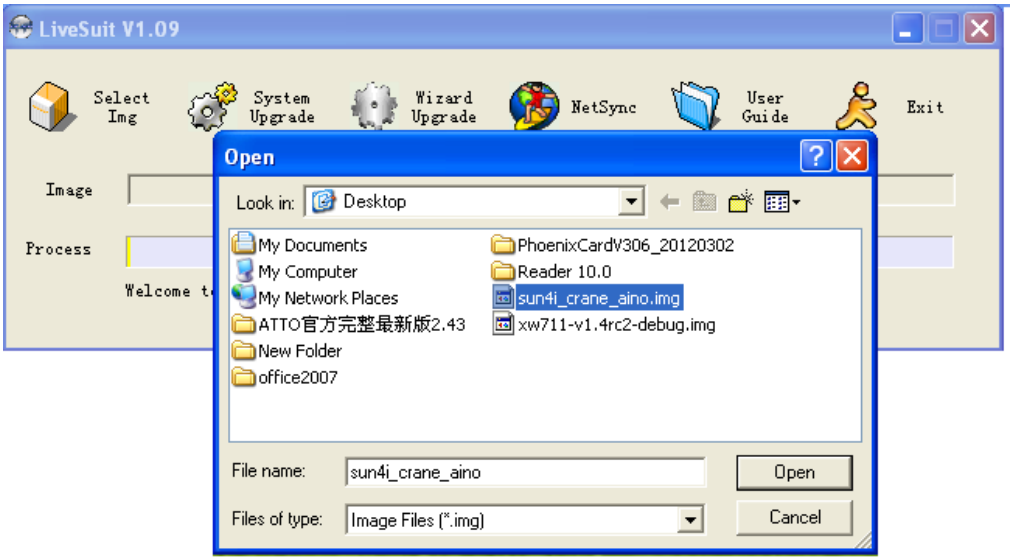


3） Start the liveSuit.exe.



Tips and Tricks: When run for the first time, a livesuit.exe shortcut will be generated on the desktop. The shortcut key on Windows platform is “CTRL+SHIFT+R”;

4） Click “Select IMG” to select the target firmware:



5） Connect a device to the PC via USB OTG, and the firmware will be flashed to the device. Since the method of each customized board entering upgrade mode varies, please consult related program personnel for details.

4.2. Card-Based Firmware Flashing

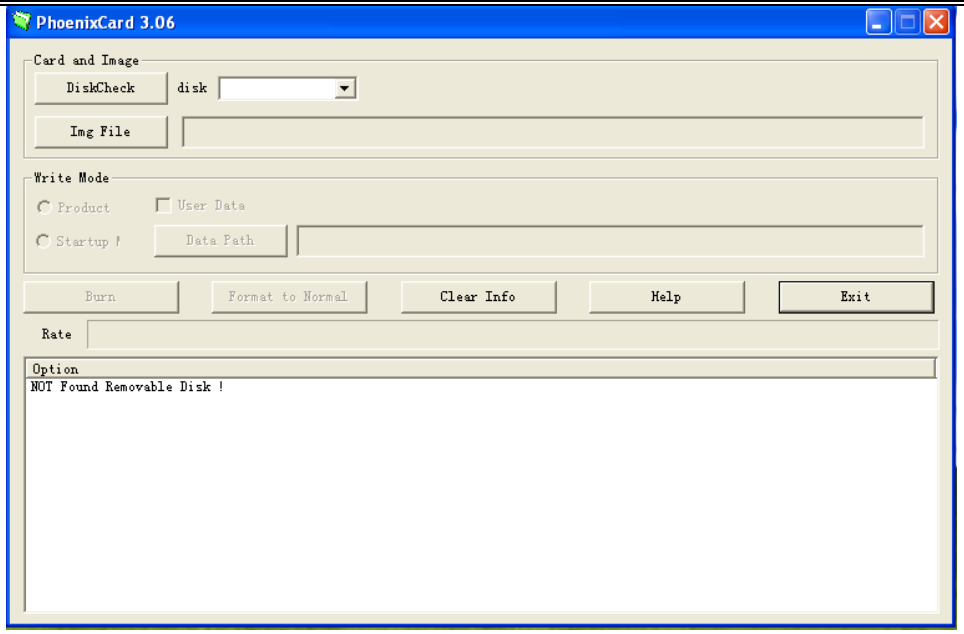
PhoenixCard can be used to flash firmware to pluggable memory cards, **startup mode** or **product mode**.

4.2.1. Startup Mode

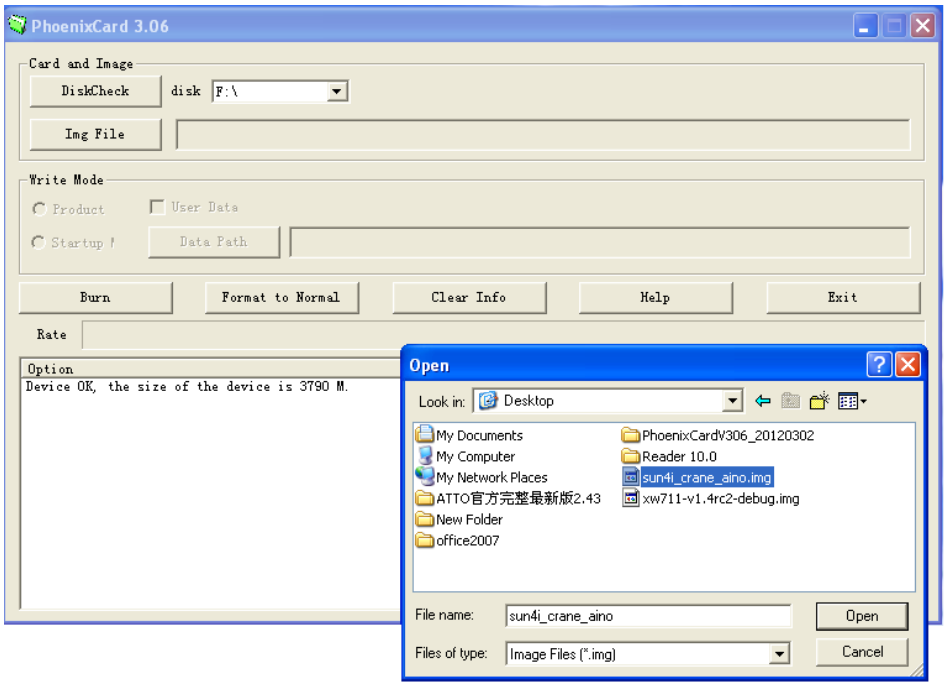
If firmware is flashed to the memory card in **Startup Mode**, it means that card will be able to be used to boot a system.

The PhoenixCard is in directory *lichee/tools/tools/tools_win*, and is executable instantly after extraction.

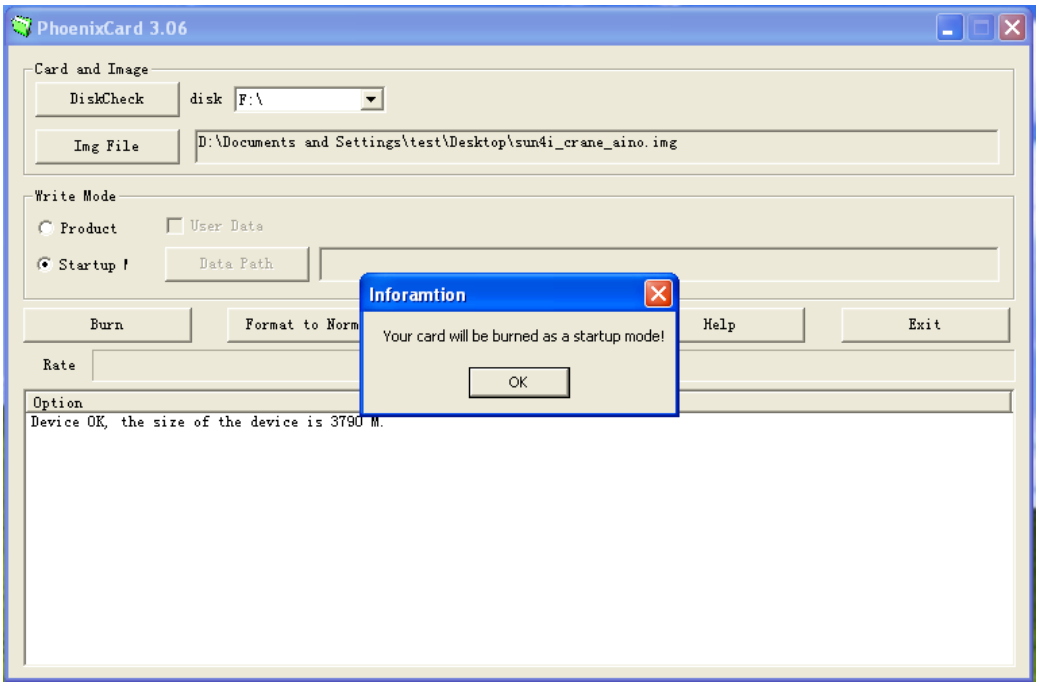
1） Extract and run PhoenixCard.exe, and you can see following interface:



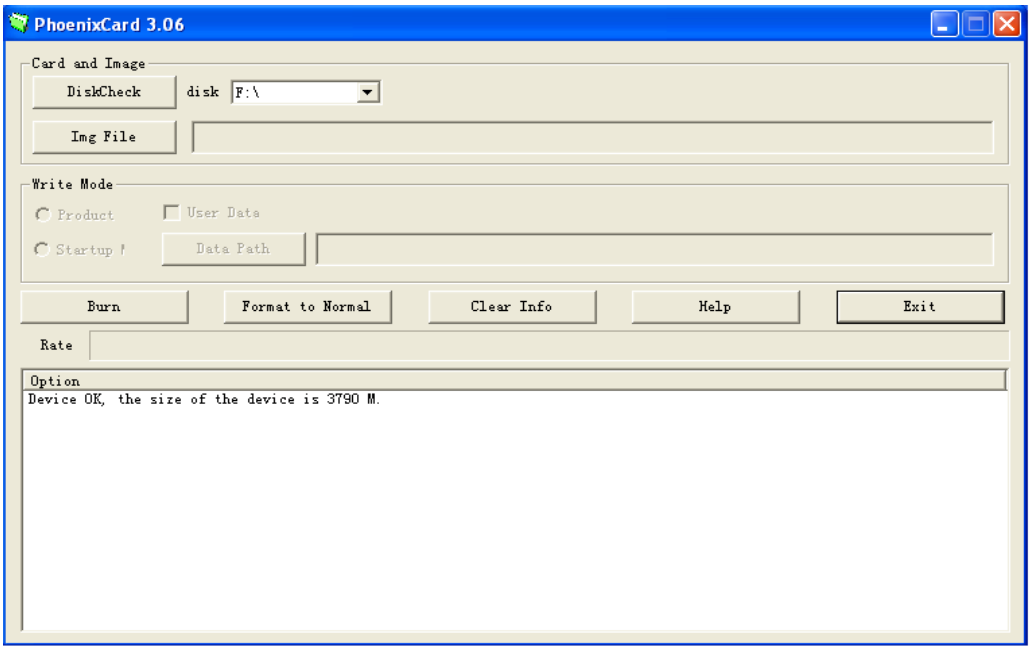
2） Click the “IMG File” button to select the target firmware:



3） Select the “Startup Mode”:



4） Insert the card (the card can be inserted anytime before clicking “Burn”):



5） Click the “Burn” button, and all buttons remain gray before the firmware flashing completes.

6） After the firmware flashing, click “exit”:

7） Insert the card to the device card slot, start or restart the device, and then it will work normally.

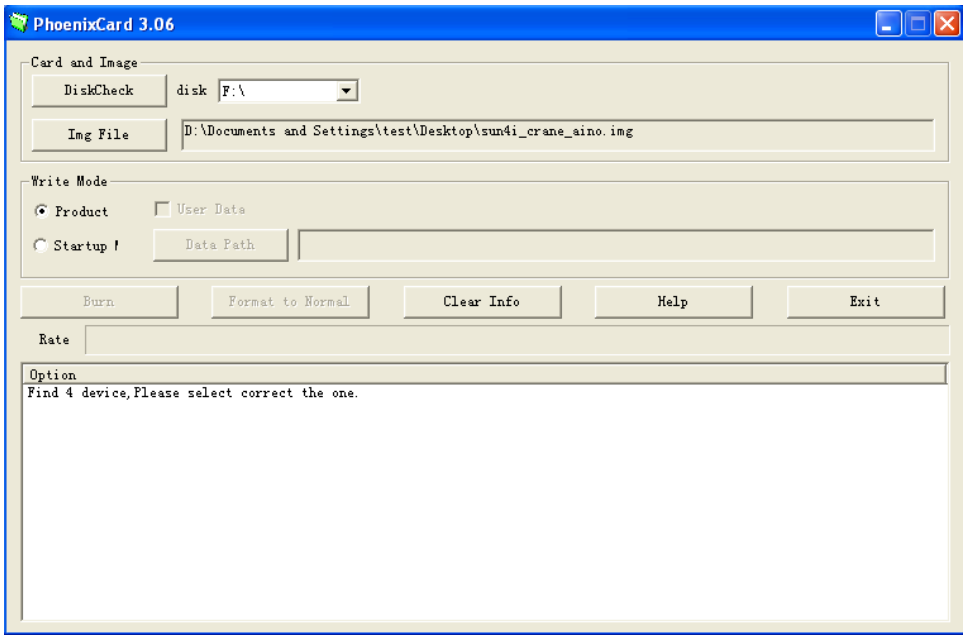
4.2.2. Product Mode

If firmware is flashed to the memory card in **Product Mode**, it means that card will be able to be used

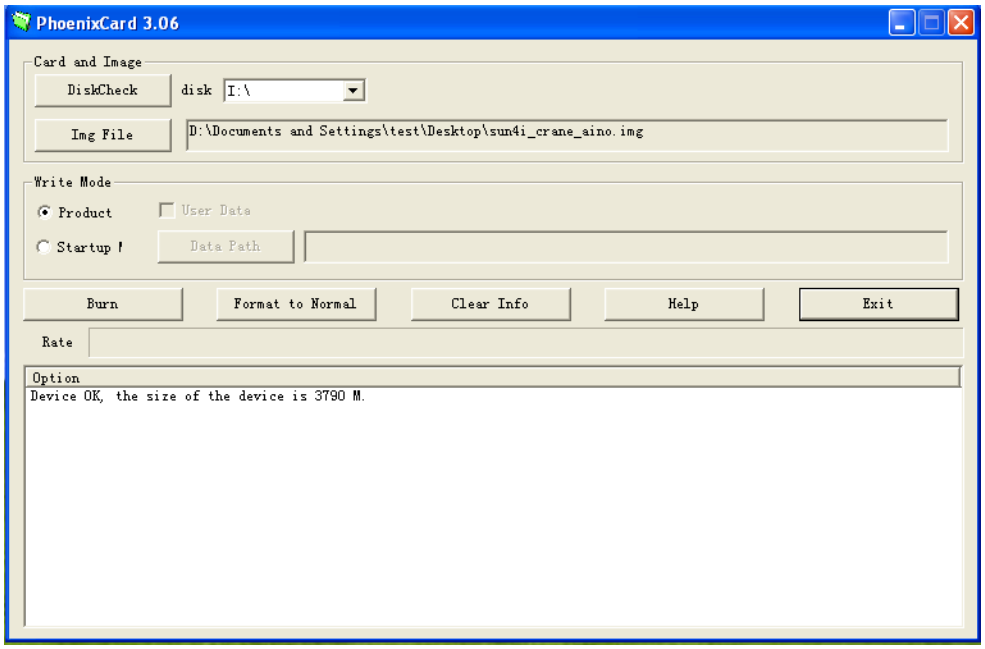
repeatedly to flash firmware to devices.

Detailed procedures are illustrated as below:

1） Insert an SD/MMC/TF card to PC via the card reader, and PhoenixCard will automatically recognize the disk letter and card capacity, which also can be accessible by manually clicking “Disk Check”. To guard against misoperation with other removable storage devices, you will be prompted to disconnect all other devices before the firmware flashing, as shown below:



2） Click the “IMG File” to select the firmware to be flashed, and select “Product” as the Write Mode：



3） Click the button “Burn” to start the firmware flashing, and the progress bar will indicate the mount progress.

4） When the bar is full, the firmware flashing is done, and that memory card shipped with firmware can be used repeatedly to flash firmware to devices.

5. Declaration

This *Memory Card Boot Solution for A10* is the original work and copyrighted property of Allwinner Technology ("Allwinner"). Reproduction in whole or in part must obtain the written approval of Allwinner and give clear acknowledgement to the copyright owner.

The information furnished by Allwinner is believed to be accurate and reliable. Allwinner reserves the right to make changes in circuit design and/or specifications at any time without notice. Allwinner does not assume any responsibility and liability for its use. Nor for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Allwinner. This documentation neither states nor implies warranty of any kind, including fitness for any particular application.