

Porting Android on to Different Platforms/Silicons

1. We know that Android is a software platform based on Linux. Most of the porting jobs are actually the porting of Linux to various platforms/silicons. So, our first task is porting the Linux.
2. If the original operation system is Windows Mobile or Window CE. There is a good tool named HaRET, which runs under WM/WIN CE. It can directly load linux kernal from Windows Mobile. We will make good use of it.
Here is the HaRET homepage: <http://www.handhelds.org/moin/moin.cgi/HaRET>
3. With this good tool which can load Linux zImage file from WM; our next task is just to make a zImage file which can run on our current silicon platform say OMAP or PXA 3xx or Qualcomm's MSM series. Due to the lack of driver support for the hardware of current platform, the zImage file compiled from the official Linux kernel source code, which is downloaded from kernel.org, can not be run on current platform.
However, we could find a close start point. Please visit here:
<http://www.handhelds.org/moin/moin.cgi/GettingHandheldKernels>
The handheld project maintained a version 2.6.21 based Linux kernel tree which supports a lot of PDA devices such as HP/Compaq, HTC, DELL, etc. Also we can find some of the Asus Models are supported. But not including CURRENT PLATFORM.
This version of Linux kernel is just our start point.
4. Based on the handheld Linux kernel source, we can write our own drivers supporting CURRENT PLATFORM by referring to the drivers of similar devices such as Asus A730, Asus 696, HTC magician, etc. Most of the hardware chip models used by these devices are actually the same ones.

But how could us know what type of chips our CURRENT PLATFORM uses? If you are brave and rude enough, you can disassemble your CURRENT PLATFORM to have a thorough looking at the internal structure. However, if you think you should always be nice to your beloved baby CURRENT PLATFORM, just refer to the pictures I took:

<http://sites.google.com/site/siteofhx/Home/android/current-platform-hardware>

5. Now we know the chip types, but we have no idea how these chips are connected with the main CPU. I don't think we could get a circuit diagram from Asus. So we have to figure it out by our own.

Fortunately, we already had a powerful tool, HaRET. If you have studied carefully with the previous steps, you must already have the answer in your mind. Yes, by means of HaRET, we can easily get to know the GPIO numbers corresponding with each of the component chips, such as Keypad, LCD, touch screen, MMC card, Bluetooth, WiFi, etc.

Just try and make notes. With this information, we can begin writing our own driver.

6. OK, after getting all the above prepared, it's time to begin. Just bend over and work hard, coding, compiling, debugging, copying and pasting. Use all your capability, do whatever you can. Finally, you get a zImage file which can run on your CURRENT PLATFORM.

Of course, you'd better use this cross comiler toolchain:

pdfMachine - is a pdf writer that produces quality PDF files with ease!

Get yours now!

"Thank you very much! I can use Acrobat Distiller or the Acrobat PDFWriter but I consider your product a lot easier to use and much preferable to Adobe's" A.Sarras - USA

http://www.codesourcery.com/gnu_toolchains/arm/portal/package3696/public/arm-none-linux-gnueabi/arm-2008q3-72-arm-none-linux-gnueabi-i686-pc-linux-gnu.tar.bz2

7. However, with this zImage, you can only enter into a black white command line interface. You have no idea whether your keypad, LCD and touchscreen can work properly. To work around this, go to this page:

<http://familiar.handhelds.org/releases/v0.8.4/install/files/ipaq-pxa270/>

Download a rootfs from the above page. Extract the entire file in the rootfs file to a spare SD card formatted as EXT2 file system. Now, you are able to boot up your Linux kernel, run the initial script in the rootfs and enter into the GPE or OPIE graphic interface, where you can verify whether your keypad, LCD and touch screen are working correctly.

If not working right, you should bend over again and don't head up until you get all of them done.

8. With previous hard working, we have conquest the porting of the first Linux. Now let's get some more.

The Linux version used by Android SDK 1.0 is 2.6.25. What we just finished with the porting is 2.6.21, how different are these two version? It's time to take out another powerful weapon - Meld Diff Viewer. Armed with this weapon, all following jobs will be so easy.

Download an official 2.6.25 Linux source from Kernel.org; compare it with the handheld 2.6.21 on which you have been worked. Is it terrible to find so much difference? Don't worry, we will work it out.

Our previous work is accomplished on the basis of the handheld maintained version. Which make us feel a little bit like climbing up by stepping on other's shoulders? It's all right; at least we have learned how to climb. Now let's back to the ground, and climb up by ourselves.

Download an official 2.6.21 source code from Kernel.org, compare it with your handheld 2.6.21, merge the missing drivers from handheld 2.6.21 to official 2.6.21. Watch the direction and don't merge all the different files, pick only those needed by CURRENT PLATFORM. By this way, you will get a clear view of what should be done to make an official linux source running on your CURRENT PLATFORM.

After getting the merging completed, compile and debug this official 2.6.21 version, let it run and enter into GPE or OPIE GUI.

9. Now let's re-compare the official 2.6.25 and the just-finished official 2.6.21, see, no much difference, isn't it? In the same way, merge the missing drivers from 2.6.21 to 2.6.25, compile and debug, make 2.6.25 run.

10. It's time to get real! Let's fight with Android.

First, download the linux source used by Android from:

<http://code.google.com/p/android/downloads/list>

pdfMachine - is a pdf writer that produces quality PDF files with ease!

Get yours now!

"Thank you very much! I can use Acrobat Distiller or the Acrobat PDFWriter but I consider your product a lot easier to use and much preferable to Adobe's" A.Sarras - USA

Second, download the Android SDK 1.0 from:

<http://code.google.com/android/download.html>

Then, extract the Android rootfs file system according to the instructions described here: <http://discuz-android.blogspot.com/2008/01/extract-google-android-file-system.html>

Now the only thing we lack is just the Linux zImage for Android

11. Compare the just-finished official 2.6.25 with the just-downloaded Android linux 2.6.25, find the difference between them, and merge the missing stuffs regarding Android from Android linux 2.6.25 to official 2.6.25. Watch the direction. Among the differences, leave alone those contents in relation to QEMU, Goldfish and yaffs. They are no use to us, don't merge them. Actually you will find that only few changes on official Linux were made by Google to suit the needs of Android.

More detailed discriptions regarding this step please refer to:

http://elinux.org/Android_on_OMAP

After compiling, we get another zImage. Debug it, let it can successfully run into the initial script in the Android rootfs.

12. Entering into the Android GUI is not as easy as entering into the GPE or OPIE GUI.

Because two special features of the LCD frame bufrer driver are required for Android's GUI to work smoothly. These two features are double buffering and pan function. Please refer to this page to modify your frame buffer driver:

<http://androidzaurus.seesaa.net/article/105551643.html>

After getting the frame buffer done, you will be able to see the Android Robot flashing in the screen, and then enter into the exciting Android GUI after a while.

13. At this moment, the touch screen is not working yet! No touching, how could we enjoy Android? So please hold on, let's finish the last step.

The reason why the touch screen does not work is that the WM97XX touch screen driver in the official release does not send BTN_TOUCH event to high level layer when the screen is touched. So Android give no response no matter how strong we hit the touch screen. Knowing that reason, you should know what to do, don't you?

Besides, the X coordination sent by the screen driver is correct, while the Y coordination is upside down. To solve this issue, refer to:

<http://androidzaurus.seesaa.net/article/90045743.html>

14. References:

[1] Google Android homepage - <http://code.google.com/android/>

[2] Benno's engineering Blog - <http://benno.id.au/blog/>

[3] Android on Zaurus - <http://androidzaurus.seesaa.net/article/74237419.html>

[4] Omegamoon Blog

[http://www.omegamoon.com/blog/index.php?m=04&y=08&d=23&entry=entry080423-212550&cat](http://www.omegamoon.com/blog/index.php?m=04&y=08&d=23&entry=entry080423-212550&category=1)
egory=1

[5]AndroidPortingOnRealTarget

<http://wiki.kldp.org/wiki.php/AndroidPortingOnRealTarget?refresh=1>

-Sayiram K N

pdfMachine - is a pdf writer that produces quality PDF files with ease!

Get yours now!

"Thank you very much! I can use Acrobat Distiller or the Acrobat PDFWriter but I consider your product a lot easier to use and much preferable to Adobe's" A.Sarras - USA