Article

Sachin Patil* and Srikant Patnaik**

*Linux System Administrator in Indian Institute of Technology, Bombay
**Research Assistant, FOSSEE, IIT Bombay

GNU/Linux on Aakash

Introduction

In the January 2013 issue of Communications of CSI, the article "Genesis of Aakash" had explained the events leading to the creation of Aakash this has been explained by Moudgalya, Phatak, Sinha, and Varma. In this article, we will explain the work that we undertook to port GNU/Linux in native mode on Aakash.

Android is a great platform, it's free, and easy to learn. Most of us will acknowledge the use of Google's Android on Aakash. The reason is that Android is not designed to run GNU apps, although it is based on Linux Kernel. This means that one has to rewrite all useful apps on a new platform, using only Java programming language. Beside these restrictions, Android also tracks user activity, and apps may contain ads which are difficult to manage.

The first version of Aakash that we worked on came with the Android Ice-cream sandwich version. It looked nice with a visually pleasing user interface. It had multiple desktop support, efficient menu applications, perfectly suited for any touch based device. On scanning through, we came across the picture gallery, calendar, messaging, contacts, clock, etc. These applications come by default with any Android device. The first question that came to our mind was, 'Is this what will go to our students? What will they do with it?' Most school going children would be unaware of how to use these apps, and instead would play around with drawing tools, games, and other items, which they are familiar with. Sadly no Android device comes with preinstalled educational applications by default.

Furthermore, Android was not intended to serve educational purposes. It was created to serve as a mobile operating system. An operating system with GUI specially designed for calling and messaging, which improved as it evolved. Several applications in android were written by developers across the world. With time, its user interface was optimized, making it easier to use. Until now, Android was largely used as an entertainment operating system. It is well suited for those who just want their work to be done without knowing what goes within. In contrast, Aakash is specially meant for education: we don't want our student to stop with playing games; nor do we want to restrict them to the development of another `Angry Birds' kind of game. We want them to learn, read, write, and carry experiments on their device. This device should be considered equivalent to any desktop we use today. We aimed to give them a full fledged device to help, play, and experiment without any limitations.

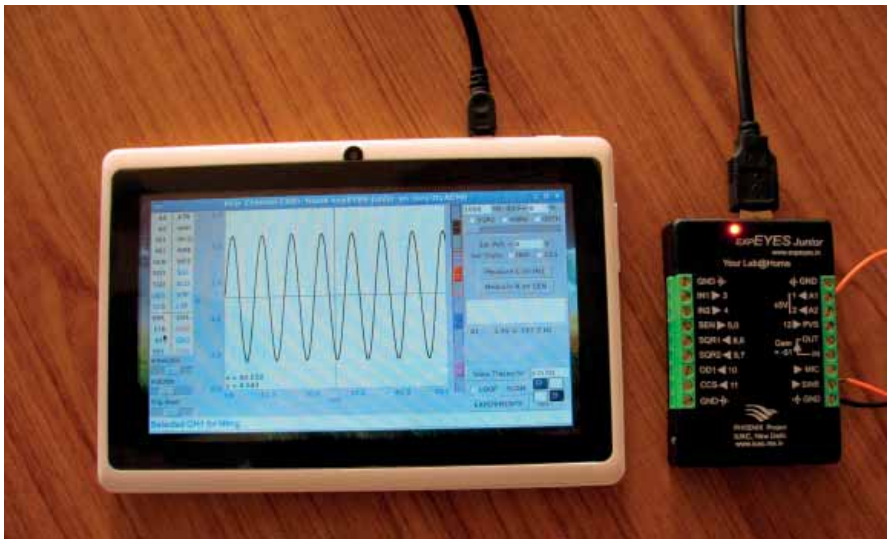
Why GNU/Linux (where Android lags)

Android uses the same Linux kernel optimized for embedded devices and excellent memory management. Though it has an easy to use GUI, it is generally not suited for everyone, especially students, as explained above. This serves strong motive to port GNU/Linux on Aakash. Although GNU/Linux uses the same Linux kernel, its file system differs completely. It gives us complete freedom to explore every part of software as well as hardware. The best part is, the GNU applications allow the curious learner to read the code and find out the way the applications work. If one is dissatisfied with the application, one is free to download the source code and modify it accordingly. He or she can even go on and share their modified code with the community, provided they acknowledge the original authors and attach the same GNU license with the code. This is where Android lags behind Aakash: most of the popular free apps are closed source in nature.

Moreover, because of the licensing restrictions, it is not possible for us to distribute useful Android Apps with Aakash, since we have to get permission from the creator of each App individually. Each one may want different agreement forms to be signed. A bigger problem is that most creators of popular Apps are difficult to locate and hence, the mails to them will go unanswered. GNU/Linux, on the other hand, is especially designed for such campaigns.

Porting

On exploring the hardware of Aakash tablets: we realized that this relatively new SoC from All-winner has support for GNU/Linux, which although limited, was sufficient enough to start our work. In pursuit of a development board needed to start our work, we looked around and finally decided to open the device itself. We asked the vendor for pin numbers that were needed to obtain the transmit data(Tx) and receive data(Rx) pins out from CPU. This information is required for debugging. With the help of a USB to serial converter, we connected the pins to a computer using an USB port. Our hardware team managed to get those



pins out, after which and our development device was ready. Without the serial out, it is difficult to track the booting process.

We had two choices, either to let the GNU/Linux boot from its internal memory(NAND flash) itself or to let the complete OS boot from an external SD-card. Fortunately the All-winner chip has a facility to boot the OS from the SD-card. Interested learners can boot GNU/Linux from SD-card without touching any part of Android.

We started compiling the boot loader. The boot-loaders on embedded system are different, and in Aakash the complete OS has to boot from the SD-card. Then we went on to compile U-boot. It is the uboot binary file on SD-card, which makes the SD-card bootable. After successfully loading, the uboot finally calls on the kernel to initialize hardware. We used minicom to view all booting processes. The next major challenge was the Linux kernel. It is the most important part, as all the hardware and applications ultimately depend on it. If the kernel successfully detects all the hardware, then we can proceed further to test the file system. If not, we need to fix it by analyzing Android kernel logs, keeping in mind all the hardware and configuring the same in our present kernel. Thanks to the open source community, we found forked versions of original Linux kernel maintained by All-winner team.

We used the default cross-compiler as suggested online. It gave compilation errors and the compilation process failed frequently. This is the same cross-compiler that is readily available in Ubuntu's repository. After many trial we decided to use the Codesourcery's cross-compiler tool-chain. We have used these tool-chains in the past. To set up Codesourcery's tool-chain, one has to register before downloading its binary. After downloading, it has to be installed and a custom path to the tool chain has to be set in order to compile the kernel. We first used the Debian root file-system, which we got online. The script.bin file was not fully compatible with Aakash. It took us some time to extract Aakash's own script.bin file and to change its default parameters to make a running kernel and root file-system. Merely modifying script.bin file was not enough, some kernel modules like WiFi and touch have to be auto-loaded while booting. These changes have to be made in the file-system path /etc/modules to make them work. With all those changes, we had basic version running in a week.

On Aakash, GNU/Linux boots from micro SD-card and the file-system reside within the SD-card. Both Android and GNU/Linux operating systems are completely isolated from other. The good part is that we can access all the Android's content from GNU/Linux.

Enabling touch was a major challenge, as we have never worked on touch before. Initially when tried to interact with the tablet using touch on Debian, it didn't respond. We had to go through Android's log-cat and dmesg to identify the touch screen driver, which we found out was focal-touch(ft5x_ts). When it was enabled as a module, it worked but we had to disable the multi-touch feature in the file. Currently Aakash has three touch screen drivers, of which two work.

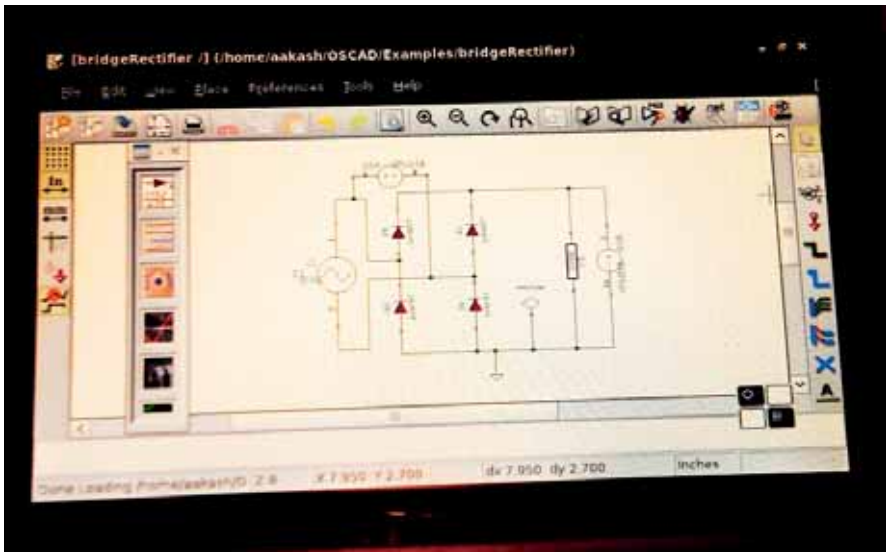
For an application like ExpEYES (explained below) and Arduino, which uses an USB-to-serial interface for interacting with the hardware, we had to enable kernel support for Communication Device Class(CDC) ExpEYES as ACM device. On the device level, it is detected as Abstract Control Model(ACM) drivers. The Linux kernel detects /dev/ttyACM0. Also for ACM to work, generic USB support should be enabled in the kernel.

On the desktop, we can pass arguments to the kernel, ask kernel to load some modules, disable misbehaving modules and so on. Similarly we can pass arguments to kernel on an embedded device using script.bin file. On Aakash, pre-customization, module loading for wireless networks, setting display resolution, loading touch drivers, etc., can be done using script.bin. Although Script.bin is a binary file, actual editing can be done by converting it to fex format.

Ubuntu as GNU/Linux Distribution

With the Linux kernel in place, now was the right time to decide upon file system . We tried Debian first, but due to lack of hard-float support in Debian Squeeze release, we dropped it. We also tried Debian Wheezy but we were uncertain about of pre-release versions. The best choice left to us was Ubuntu, which is known for being easy to use amongst newcomers and advanced developers alike. It has a great package-manager, using which one can install required packages both from command line and using graphical interface. Hence we agreed to use Ubuntu.

We started with Ubuntu-12.10 core arm hard-float distribution. When uncompressed, it consumes around 100MB only. We used the ch-root environment to configure package-



manager, basic network tools, user applications and a desktop environment.

Before putting the file-system to actual use, the compiled kernel and its modules were placed in /lib/modules directory of the file-system, where all kernel modules reside.

The next challenge was the Desktop environment. We tried Unity, KDE Plasma, XFCE, MATE, enlightenment (e17), and Gnome-3, all of which need around 120 MB RAM with some hardware acceleration, except e17. After considering the options we finally decided to go with LXDE(not Lubuntu).

The Ubuntu-core file-system includes only basic utilities and a package-manager (apt-get). Comparing with a Desktop version, it does not even have a basic networking tools like ping or root user utilities, such as sudo. Ubuntu Boot-splash screen, Desktop-Environment, screen savers, UbuntuOne sync, daemons, etc., consume a lot of memory. We compared memory consumption of each process before installing one in core file-system. For example, Ubuntu's default Desktop-Environment(Unity) consumed much more memory than LXDE. By discarding these overheads, we finally managed to boot Ubuntu in less than 50MB RAM. We also made a few customizations on open-box and gtk2.0 to make it touch friendly.

Applications

We focused largely on educational applications. With repositories in the path, one can easily install any application of one's choice. We pre-installed some popular and useful applications. The first application we installed was Onboard, to serve as the virtual keyboard. Next we installed the LibreOffice pack. Although it's a bit heavier than AbiWord, its features make it worth installing. Scilab-5.3.3 was also installed and tested. Both numerical and graphical calculations are executed much faster than on Android(https://github.com/androportal/APL-apk). More than 150 Scilab textbook companions (http://scilab.in) are now available in our latest builds. A Scilab textbook companion is a listing of code that implements worked out examples in standard textbooks. Arduino, an open source hardware with Gnoduino IDE, has also been tested and included.

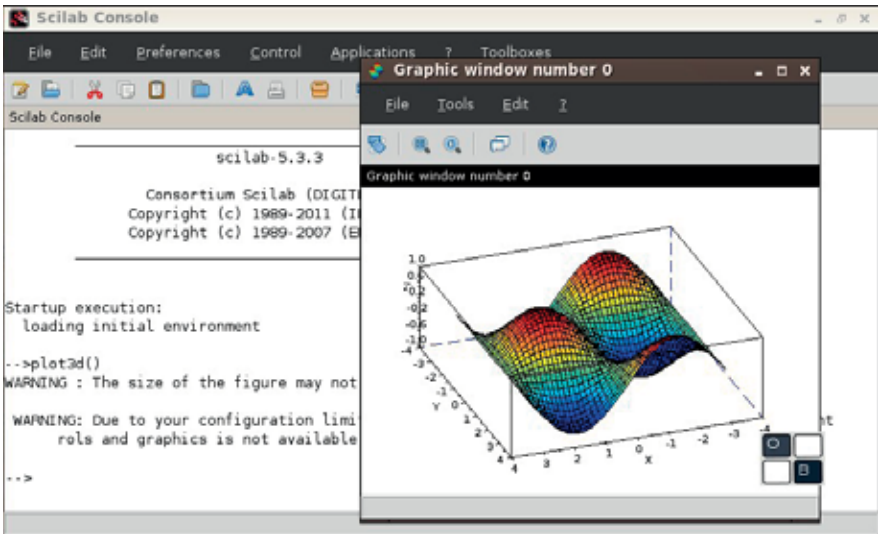
ExpEYES is a hardware and software tool for learning and exploring science experiments. It supports 50 experiments for high-school and above. For interacting with the hardware, we have a Debian package called ExpEYES Junior. This is a tablet version alternative for ExpEYES in desktop.

OSCAD is another open source EDA tool, acronym as Open Source Computer Aided Design. It has been developed using several open source tools like KiCad, Ngspice, and Scilab at IIT Bombay. Python-TKinter is used to program its front-end. Since tools such as KiCad, Ngspice, and Scilab already run on Aakash, OSCAD's installation procedure was similar to any other desktop running GNU/linux. Aakash's capability to run Electronic design tools is demonstrated by the fact that OSCAD runs on it.

We also installed iPython-notebook for scientific computing, and Mayavi2 for 3D visualization of data.

Conclusion

After these customization process, we now have Ubuntu 12.10 with Linux kernel version 3.0.57 working on Aakash. It is suitable for educational as well as entertainment purposes. For programming and development one can attach an external keyboard and mouse, if one is not comfortable with virtual keyboard. One can see all the features and application of a typical desktop computer on Aakash. With ExpEYES and Arduino working, one can perform hardware interfacing



with any other hardware. GNU/Linux on Aakash provides opportunities to experiment on a portable device. With 1 GHz processor and 512MB memory, it has the potential to run any other GNU applications. Currently we have image targeted for 8GB SD-card of which first 16M FAT partition is dedicated to bootloader(uboot.bin) and script.bin file. 1GB is used as swap file-system in case if the actual RAM gets used up. The entire file-system along with install applications and other utilities consumes around 3GB space. Approximately 3.5 GB is left free for storage and other installation to user. The capacity of the SD-card can be expanded up to 32GB.

Contributing to Project

We look forward to seeing GNU/Linux enthusiasts contribute to this project. Please visit our github page for detailed documentation on porting of Aakash. There are many open issues, such as, brightness control, sleep mode, touch drivers, etc. We have documented our work at http://androportal.github.com/linux-on-aakash/.

Aakash Application Development Competition

In January 2013 issue of Communications of CSI, we had announced a competition based on Aakash, for both Android and GNU/Linux operating systems.

This competition aimed to encourage students and individuals across the country to come up with innovative applications that could be used on Aakash. The source code of each application will be released as free and open source. The Application can be Android or GNU/linux based.

More that 1600 participants registered for the Aakash application development competition. These participants are from various engineering colleges and universities across India. We asked those participants to re-group in teams consisting of maximum 5 people, and re-submit their project proposal. On the basis of project description, we have shortlisted 140 teams, whose work will be developed further. Any updates related to competition will be posted on http://aakashlabs.org/compete.

Traditionally all applications running on GNU/linux desktop should also run on GNU/linux on Aakash. But one must ensure that the application is touch friendly and consumes minimum RAM. The Aakash team at IIT Bombay is willing to help the participants: for example, the participants: for example, the participants can send their application to us for testing.

We still have many open issues on GNU/linux port. Before contributing, we expect the participants of the competition to go through our GNU/linux porting documentation on github page http://androportal.github.io/linux-on-aakash/.

We are in need of developers who are interested in GNU/Linux system. They must have sound knowledge of Linux kernel and working of various GNU/linux distributions.

LocalWords: bootable uboot minicom online Codesourcery's Aakash # LocalWords: Aakash's WiFi NAND Bootloader dmesg multi ExpEYES USB # LocalWords: Arduino ACM linux github pre onboard LibreOffice # LocalWords: AbiWord Scilab ■

About the Authors

Srikant Patnaik He is a developer, teacher and motivator. His first contribution to FOSS came as a simple 8051 Programmer for Linux, available at sourceforge. He served as a Lecturer at Loyola academy, Hyderabad. Later joined IIT Bombay as a Research Assistant in FOSSEE project. He contributed in Porting of GNU/Linux on Aakash and also associated with Android app to run Scilab and other programming languages. His interests include blogging, designing circuits, bridging software and hardware.

Sachin Patil is currently working as a Linux System Administrator in Indian Institute of Technology, Bombay. Apart from System Administration, he has also gained some experience in Android and embedded systems. He, along with Srikant Patnaik, has ported Scilab — a software for Numerical Computation on 'Aakash', a low cost access device project funded by NMEICT, Govt. of India. He is also interested in customising GNU/linux distributions. Beside Ubuntu, his other favourite GNU/linux distro is Slackware, which he likes to work on because of its simplicity and robustness.