Linux chroot on Android • Real life SD card performance test

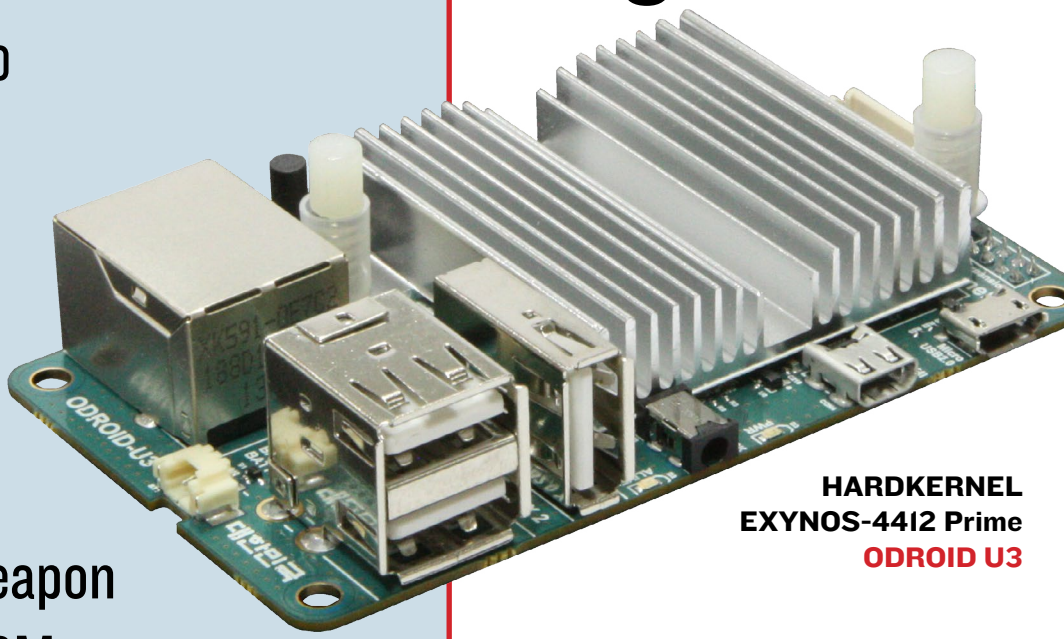
ODROID Magazine

If you like the ODROID-U2,
you'll love our new ODROID

U3

hardkernel's new weapon
for developing on ARM

And it is amazing!



HARDKERNEL
EXYNOS-4412 Prime
ODROID U3

RPi vs U3

We show you how they stand against each other

Ubuntu 13.10

Install and troubleshooting
(the easy way)

- Use C/C++/Python/Java and IDE to:
  Hack your hardware and control LEDs
- Start programming right away!
- High Performance
  Computing with multiple ODROIDs
- Meet an Odroidian
  Justin Lee, CEO of Hardkernel
- Linux Gaming
- Real-Time Mobile Eye Tracking
- Introduction to the HTML5 Video Player

A service to the world-wide ODROID and Open Source communities, Hardkernel is proud to present its newest contribution to ARM technology: ODROID Magazine, a free monthly PDF e-zine!

This cutting-edge online publication brings you the latest ODROID news, as well as featured articles from the expert community that has grown around the amazing ODROID family of micro-powerhouse computers.

Intended for all levels of expertise from beginner to guru, ODROID Magazine features definitive guides for new owners, with easy-to-follow steps in setting up your ODROID, installing operating systems and software, and troubleshooting common issues. For more technical users, each month will feature expert tips, hacker discussions, cutting-edge projects, and technical articles to explore new ways of making your ODROID even more versatile.

Hardkernel's ODROID Magazine is an ideal opportunity for our community to come together to share and contribute articles, so that everyone can be successful with their ODROID.

Each month, a series of article topics will be posted for consideration, and all community members are encouraged to send submissions in exchange for monthly rewards for those selected for publication.

The best articles are those that walk the reader through complex concepts and procedures in a simple-to-read format. At least one picture or graphic per article is required, and should be between 500-2000 words.

Interested in being hired as an ODROID Magazine staff member? Then send a PM to @odroid, @mdrjr and/or @robroy on the ODROID forums, and we will contact you with further details. There will be a small monthly compensation for those who wish to contribute each month to the design and production of the e-zine, including translations and proofreading. It would certainly be beneficial for any IT resume to be a published, active contributor to one of the most resourceful communities in the ARM world!

ODROID
Magazine

Rob Roy, Chief Editor

I am a computer programmer living and working in Silicon Valley, CA, USA, designing and building websites such as Vevo, Hi5, Dolby Laboratories and Hyundai. My primary languages are jQuery, Angular JS and HTML5/CSS3. I also develop pre-built operating systems, custom kernels and optimized applications for the ODROID platform based on Hardkernel's official releases, for which I have won several Monthly Forum Awards. I currently have 8 ODROIDs, which I use for a variety of purposes, including media center, web server, application development workstation, and gaming console.

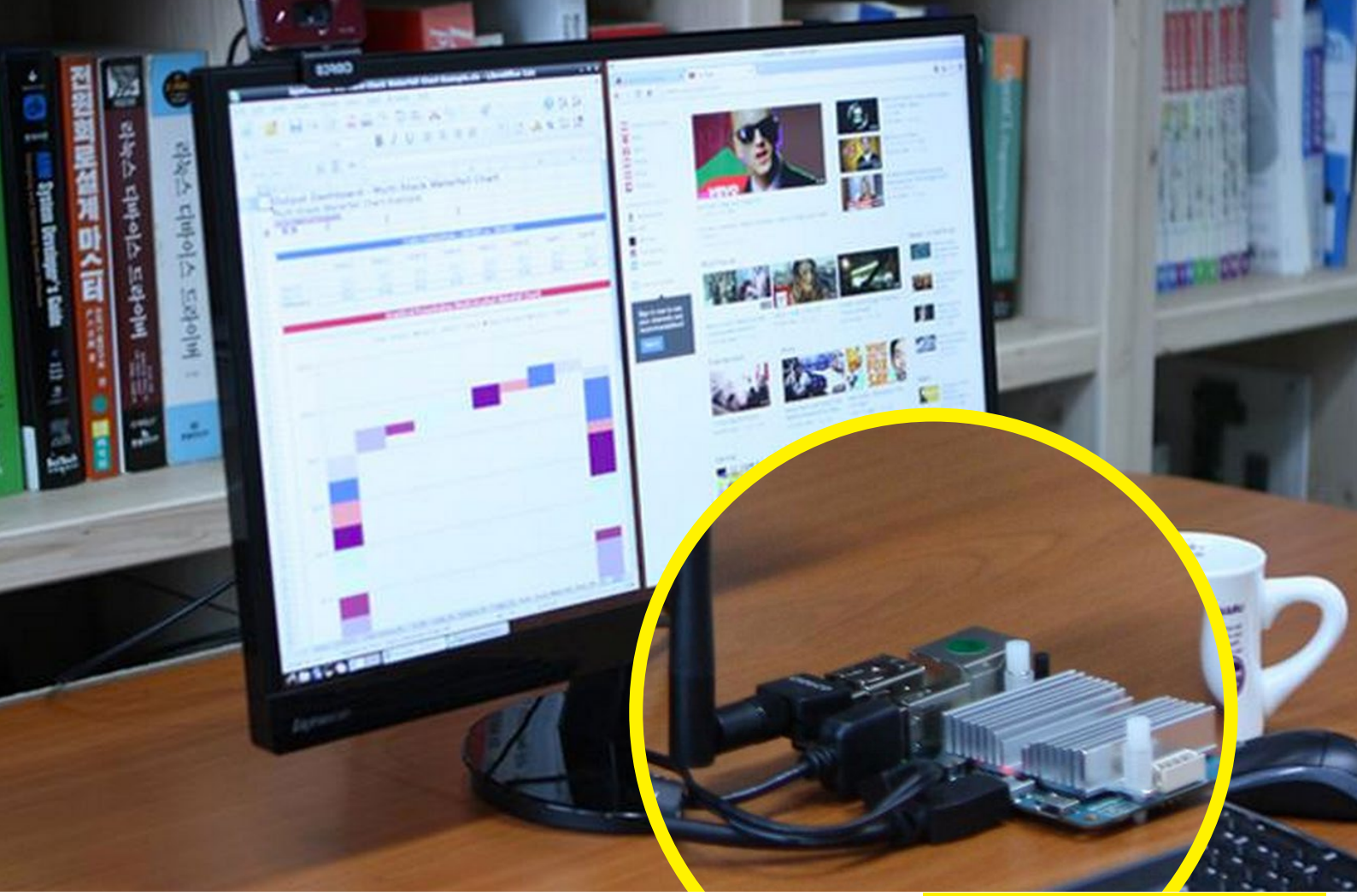
Bo Lechnowsky, Editor

I am President of Respectech, Inc., a technology consultancy in Ukiah, CA, USA that I founded in 2001. From my background in electronics and computer programming, I manage a team of technologists, plus develop custom solutions for companies ranging from small businesses to worldwide corporations. ODROIDs are one of the weapons in my arsenal for tackling these projects. My favorite development languages are Rebol and Red, both of which run fabulously on ARM-based systems like the ODROID-U2. I have deep experience with many unique operating systems.

Bruno Doiche, Art Editor

I'm just some random guy!

ODROID Magazine, published monthly at http://magazine.odroid.com, is your source for all things ODROIDian. Explore the new technologies offered by Hardkernel at http://www.hardkernel.com
Hard Kernel, Ltd.
704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815
Makers of the ODROID family of quad-core development boards and the world's first ARM big.LITTLE architecture based single board computer.
Join the worldwide Hard Kernel community with members from over 135 countries at http://forum.odroid.com.



A typical ODROID-U3 setup. A 21-inch monitor is connected to the HDMI port, and the keyboard, mouse, WiFi and USB camera are connected to 3 USB ports.

GETTING STARTED WITH THE ODROID-U3

Have you been waiting to upgrade your system because of the high prices of PC and Mac computers? Looking for a second computer for family, work or college use? Hardkernel's family of low-cost, powerful ARM devices is now even more affordable with the introduction of the powerful U3, specially priced at US$59.

With a credit-card-sized footprint and low cost, the ODROID-U3 is sometimes viewed as a hobbyist computer. However, its performance, software and ease of use make it an excellent replacement for a high-cost PC, running both Android and Linux with thousands of free and low-cost titles available. The ODROID-U3 also offers many advantages over a typical Windows or OSX computer, such as silent operation, 5W average power usage, and instant portability, since it fits in a shirt pocket.

The processor is an Exynos4412 1.7GHz Quad-core from Samsung with 2GB RAM. The size of this computer is only 83 x 48 mm with weight of 48g.

Unlike a PC, the ODROID-U3 has the motherboard, memory, sound card and even heat sink pre-assembled. Instead of using an expensive hard disk like a PC, it runs the operating system from a low-cost, micro-sized solid state SD card, like a digital camera. An SD card, as well as the faster eMMC module, can be pre-ordered with the ODROID-U3, and arrives with the popular Ubuntu operating system already installed.

Insert the SD card into the reader, connect a monitor, a keyboard, a mouse, Ethernet and power cable, and press the power button. That's all you need to do to use the ODROID-U3! Browse the web, play games, run office programs, edit photos, develop software, and watch videos right away. If you have the eMMC module, lay it onto the square area marked on the motherboard, and clip it into the board.

An efficient heatsink is standard on the ODROID-U3, which attaches to the CPU with fins to dissipate heat without a fan.

Let's play (and work) together!

The ODROID-U3 is suitable for anyone from professional software engineers building Kinect-drive robotics projects to kids learning to program with the Scratch language. Run the latest Ubuntu 13.10 and Android Jellybean operating systems for programming, learning, gaming, media center, web server, office/college work, hardware IO platform and many other applications. The powerful 1.7Ghz quad-core processor, low cost, energy efficient features and massive software library make the ODROID-U3 the perfect modern computer for work or play.

A Tour of the Board

Let's start with a quick tour of what you're looking at when you take it out of the box. It's similar to your typical PC with various additional features.

A The Processor. At the heart of the ODROID-U3 is the same processor that is used in a Samsung Galaxy S3 smartphone. This area contains a quad-core 1.7GHz system on a chip, which is built on the ARM Cortex-A9 architecture. It has 2GB of RAM stacked on the CPU.

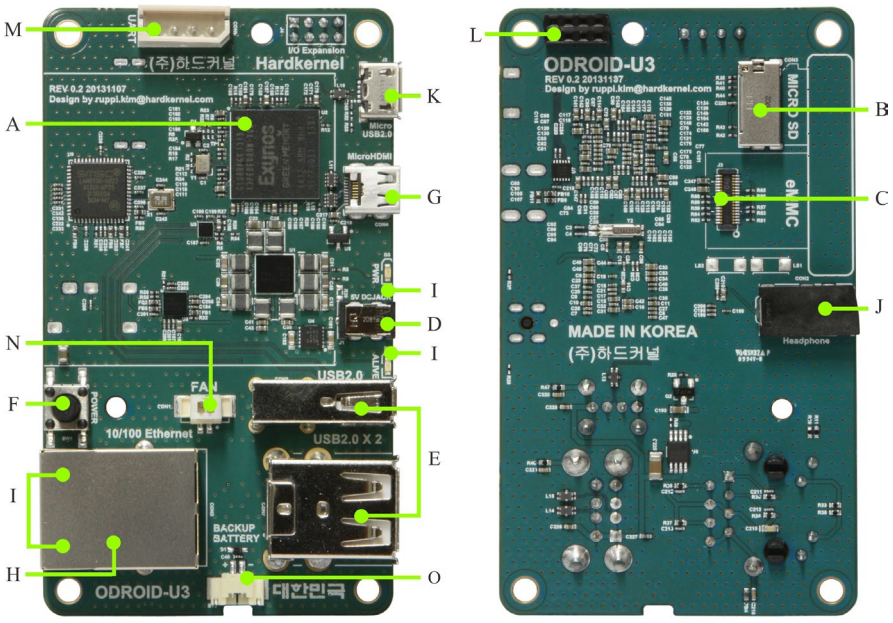
B. The Micro Secure Digital (MicroSD) Card slot. There are two different ways of storage for operating system. One is using a MicroSD Card and another is using an eMMC module. It is normally used for external storage for smartphones and digital cameras.

C. The eMMC Module socket. The eMMC storage access time is 3-4 times faster than the SD card. There are 3 options of 8GB, 16GB and 64GB. Using an eMMC module will increase speed and responsiveness, similar to the way in which upgrading to a Solid State Drive (SSD) in a typical PC also improves performance over a mechanical hard drive (HDD).

D. The Power Jack. This is for 5 volt power input, with an inner diameter of 0.8mm, and an outer diameter of 2.5mm. The ODROID-U3 consumes less than 1A in most cases, but it can climb to 2A if many non-powered USB peripherals are attached directly to the main board.

E. The USB port. There are three USB 2.0 host ports. You can plug a keyboard, mouse, WiFi adapter, or many other devices into these ports. You can also charge your smartphone via this port. If you need more than 3 ports, you can use a powered external hub if you have a peripheral that needs more power.

F. The Power button. You can turn on and off the ODROID-U3 with this button. Once you press this button to turn off, you will see a dialog box on screen to confirm the power-off request.



The four status LEDs

POWER	Red	Hooked up to 5V/2A power
ALIVE	Blue	Dim light : Boot-loader is running Solid light : Kernel is loaded Flashing : Kernel is running (heart beat)
ETHERNET	Green Yellow	Network activity light On if the network connection is 100Mbps

G. HDMI connector. To minimize the size of the board we used the Type-D micro-HDMI connector. The HDMI output supports 720p and 1080p display.

H. Ethernet port. The standard RJ45 Ethernet port for LAN connection supports 10 or 100Mbps speed. WiFi connectivity via a USB dongle is another option.

I. Status LEDs. The ODROID-U3 has four indicator LEDs that provide visual feedback (see Table above).

J. Analog Audio. This is a standard 3.5mm analog audio jack. The jack has 4 poles to support a mono microphone input via the headset for Android smartphones.

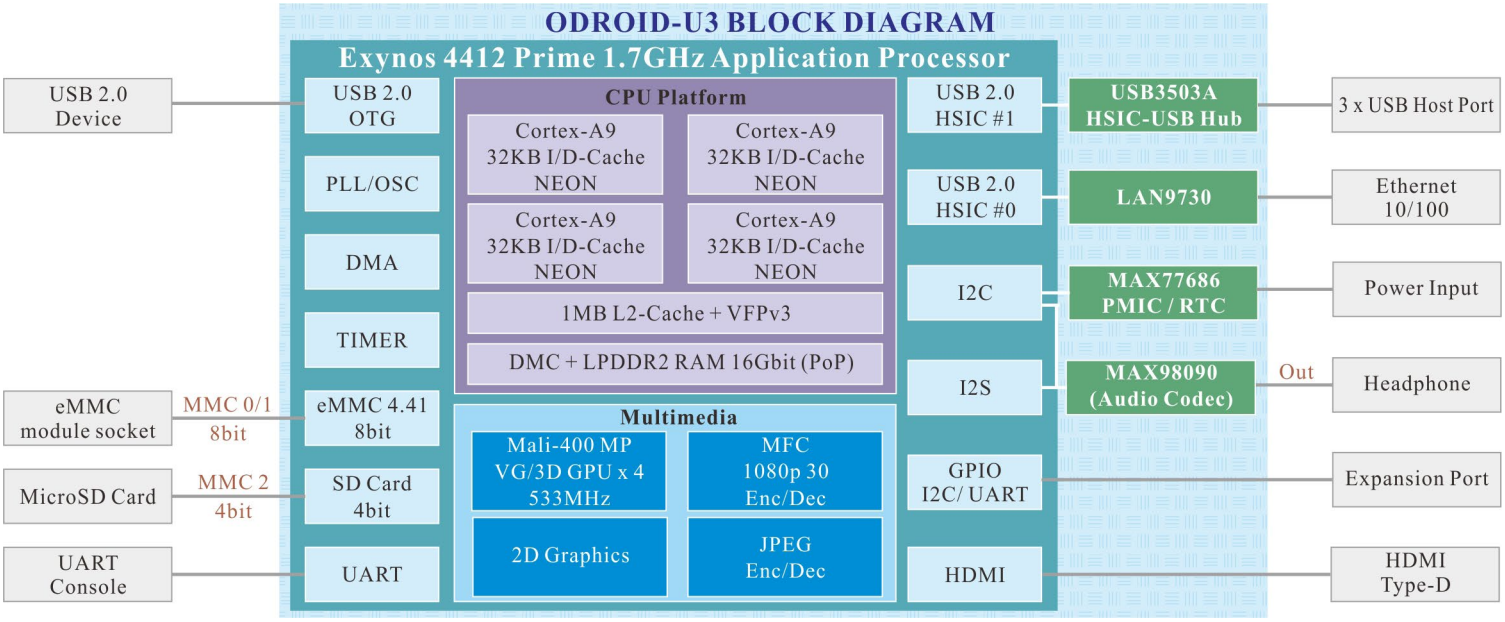
K. Micro USB Connector. This is the standard Micro USB device-only connector. You can use it with Linux Gadget drivers with your host PC, which means that the resources in the ODROID-U3 can be shared with typical PCs. It cannot be used for power input.

L. General Purpose Input and Output (GPIO). These can be used for IRQ/I2C/UART for electronics and robotics. The IOs can be expanded for many more PWM/ADC/GPIO ports.

M. Serial terminal Port. Connecting to a PC gives access to the Linux console. You can see the log of the boot, or to log in to the U3 to change the video or network settings. Note that this serial UART uses a 1.8 volt interface. We recommend the USB-UART module kit from Hardkernel.

N. PWM Output for Cooling Fan. If you run U3 in a room with high ambient temperature, a low-profile cooling fan might be helpful. The PWM output increases/decreases the speed of fan proportionally to the CPU temperature.

O. RTC (Real time clock) Backup Battery Connector. If you want to add a RTC function for logs or keeping time when offline, just connect a backup battery. RTC is implemented on ODROID-U3.



A block diagram showing the architecture of the ODROID-U3.

The Peripherals

Now that you know where everything is on the board, you'll need to know a few things about the proper peripherals (some are shown in Table 1-2) to use with the ODROID-U3.

Along with these peripherals, you also need a monitor, keyboard and mouse. Any USB wired/wireless keyboard and mouse are compatible with the ODROID-U3. The monitor or TV must support 720p (approximately 1280x720) or 1080p (approximately 1920x1080) resolutions.

A. A power supply. This is the most important peripheral. You should use a 5V/2A adapter. Plug inner diameter is 0.8mm and outer diameter 2.5mm with center positive and outer negative.

B. A MicroSD Card. If not using an eMMC module, you'll need a recommended 8GB Class 10 or higher MicroSD card. Lower class versions will affect performance. Hardkernel's official MicroSD Card has Ubuntu pre-installed, and can be booted immediately after opening the box..

C. An eMMC Module. If you want higher performance, choose the eMMC module. It is much faster than the MicroSD option and also has Ubuntu pre-installed.

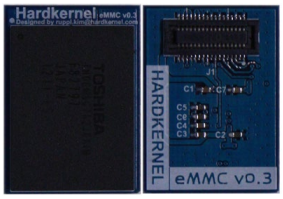
D. An HDMI to micro-HDMI cable. You need this cable to connect to a monitor. You also might need an appropriate adapter for a VGA-only monitor as the ODROID-U3 doesn't have VGA output. Hardkernel's HDMI cable is recommended.

E. WiFi USB dongle. You can use a WiFi USB dongle instead of wired Ethernet jacks. We recommend the RTL8188CUS-based WiFi adapter.

F. Case. To protect your ODROID-U3 from unexpected damage, a protective case is recommended.

Power Supply

MicroSD Card



eMMC Module

HDMI Cable

WiFi Module

USING ODROIDS IN HIGH PERFORMANCE COMPUTING (HPC)

ARM: HEAD AND SHOULDERS ABOVE THE COMPETITION.

Kurt Keville, MIT
@ http://meegs.mit.edu

Why HPC?

The modern Datacenter uses far too much electricity and air conditioning to run efficiently. An ARM-based Internet Service Provider can deliver web pages for substantially less power than conventional architectures (`http://tinyurl.com/ApacheOnARM`). At the same time, ARM cores are rapidly being adopted by the scientific Datacenter community because ARM devices compute faster in floating-point math intensive operations, for a fraction of the energy costs, and have an architectural roadmap of even more performance per watt to come. There is also a corresponding growth in interest in HPC (High Performance Computing) and its uses in broader domains from the ARM developer community. In academia, there is also High Performance Extreme Computing (`http://www.ieee-hpec.org`) and IEEE Supercomputing, which were demonstrated at the recent SC13 conference in Denver, Colorado. Additionally, IEEE Cluster, which was held in Indiana in 2013, has seen a marked rise in ARM-centric publications.

The majority of modern supercomputing centers have thousands to tens of thousands of cores dedicated to their particular processing needs. Any time a programmer can run an application at an improved performance ratio (per watt, dollar, or square meter) is a win to the Datacenter stakeholders as well as the computationally scientific domain customers they support. This is increasingly the case for situations where applications must be run multiple times, and where multiple applications share resources in HPC, a lot like the cloud and a throwback to old-style timesharing.

Why ARM?

The ARMv7 architecture has proven to be up to the challenge of HPC in a number of ways that previous ARM architectures were not. One might use ARM despite energy efficiency as the technology represents a growth path for fast embedded computing. When a Datacenter is composed of over 10,000 cores, considerable advantages are realized through incremental improvements. These small changes can add up to significant savings in space, power, and cooling. When memory is shared between the CPU and the GPU on ARM SoCs (System on a Chip), double the SIMD (Single Instruction, Multiple Data) extensions on Cortex-A15 NEON GPU, and considerably larger memory access potential, benefits are realized at the place we need them most; where the application and data sets meet on-die. With growing acceptance of GP-GPU (General Purpose GPU) computing and expansion of HPC-type applications based on big data apps, the fast computing modes of ARM are relevant in more ways than ever, with a technology path towards an ever-expanding share of HPC.

Why ODROID?

Today, ODROID has an Exynos family processor and at least 4 ARM cores. The upcoming Exynos5 series has 8 cores, 4 of which are ARM Cortex-A15. Hardkernel, as well as the RunTime Computing Solutions research consortium, have demonstrated substantial power and performance improvements of the XU in comparison to other contemporary architectures. With Hardkernel's ambitious release schedule of new technologies, adopters of this platform follow Moore's law and ARMs rollout efficiently, allowing them to join the wave of newer, better, lower cost, higher performance systems as they emerge with meaningful upward compatibility.

What's more, Cortex-A15 wins on most ARM-HPC benchmarks (for instance, the NAS Parallel Benchmarks at `http://tinyurl.com/ODROID-HPC`). The RunTime Computing Solutions team has recently demonstrated pivotal advantages of the A15 over the A9 on the HPCC Challenge, the preferred benchmark for HPC (`http://hpcchallenge.org`). This test uses just the A15 cores on the Exynos 5410 and maintains them at maximum capacity during the test (`http://tinyurl.com/ODROID-LINPACK`) which is not a completely equitable comparison. However, meaningful information can still be gleaned from it; XUJessie is twice as good as U2Whisper in G-HPLINPACK, the first test.

Conclusion

Today we can run many HPC applications on ODROID, and as the upward pressures of energy efficiency cause industry professionals to rethink Datacenter design, progressively more centers will adopt these architectures. The future is bright for designers as the push towards exascale computing ushers in a new and exciting theme in embedded SoC technologies.

INSTALLING AN OS ON AN ODROID

CHRONICLES OF A MAD SCIENTIST

Bohdan Lechnowsky
@ bo@respectech.com

As you're sitting in your semi-darkened laboratory, working on your next project for world domination which you hastily scribbled down on a slightly used napkin from last night's meal, you jump at the sound of a klaxon. Although at the time you installed the klaxon you thought it would be a fitting way for a mad scientist to be alerted that an intruder/visitor was at the door, you now reconsider as you rub your burnt hand. You hadn't considered the possibility of being startled while using your laser ray soldering iron.

You arrive at the front door, but the perpetrator of the incident that initiated the flesh searing is no longer apparent. As you turn to check your surveillance footage and activate your laser perimeter system, you notice a small brown box by the door labeled "Hardkernel". Your heart immediately jumps with excitement as you pick up the diminutive package, one that you know harnesses powers the world has only begun to realize.

During the elevator trip down to your subterranean chamber, your mind races with all the devious possibilities. Hurriedly, you rip open the package while you glide to the computer portion of the lab. It is a trivial task to plug in the micro-HDMI cable into your super-sized LED monitor, a wireless USB keyboard/mouse adapter into one of the several USB ports, and the power adapter from a switched outlet into the ODROID unit. You think, "I could use the plain outlet, but I prefer the switched one." One day, when you achieve world domination, you'll be able to afford a minion to whom you can yell, "Flip the switch!!" Until then, you flip the switch yourself and emit an evil laugh as the unit lights up.

Quickly, you realize there is a problem! Nothing appears on the screen!! Immediately, your eyes twitch from side to side as scores of names flow through your mind while you consider which nemesis would be clever enough to hinder your devious plans in such a way! As you look for obvious physical clues, you notice someone has stolen your eMMC card! Then, you remember you never purchased an eMMC card (or a microSD card, for that matter)!!

As a mad scientist, you instinctively know that eMMC cards are much faster, but require an adapter to interface with your computer and are also harder to procure than microSD cards, which are ubiquitous, inexpensive, and available at the drug store just down the street. (Your instincts are backed up by hard data located elsewhere in this issue of ODROID Magazine.) "No matter," you say to yourself, "the method is the same for either."

"Next," you ponder, "should I use Windows or Linux to burn the image?" While you weigh the pros and cons, you initiate downloading your operating system of choice from http://dn.odroid.com. Now that you have your operating system image downloading, you scribble the following on the back of one of your abandoned world-domination plans:

You know that the best domination schemes requires getting your hands a little dirty, so you're happy to do the above. Of course, being a mad scientist, you choose the command-line Linux method as it fits your mad scientist profile better.

You take the card out of your workstation, power off your ODROID (flip the switch the OTHER way), insert the card into the ODROID, and yell "FLIP THE SWITCH!!" to nobody in particular, this time knowing you are well on your way to launching your next world-domination scheme!

Some consternation comes across your face as you see only a black screen for a short while, but later find this is normal as the ODROID doesn't have console mode output. The ODROID happily displays output once the full graphics display is initiated. At this point, you enter the super-secret passphrase (try "odroid" for the username and "odroid" for the password if you are stumped), and your gigantic LED Plasma display comes to life.

ON TO RULE THE WORLD!

Windows

Get the Win32DiskImager.exe for ODROID v1.1 which is available at http://www.hardkernel.com/?b144. Windows should automatically open the .zip file after you download and double-click on it, allowing you to copy the .exe file onto desktop or another location. The advantages of this version over the version available at Sourceforge.com are that it allows wiping the flash memory before writing, and writes and verifies the data while writing to the card. Accordingly, this makes the imaging process take about twice as long (three times as long if wiping first).

The download file may be compressed in .xz format. To decompress, you need to download 7-zip or a similar utility from the Internet. 7-zip is available at http://www.7-zip.org/download.html. If you use 7-zip, you need to open the File Manager component and navigate to the location of the download, and then open the archive and save out the contents.

The "Device" field is empty if you start Win32 Disk Imager before inserting the card. Remember not to do that!. Once you save out the .img file, make sure my microSD/eMMC card is inserted into the USB port.

Next, simply open the Win32DiskImager.

If you want to wipe any previous data off the card, remember to wipe it first. Then, click the "Browse" button, find my .img file, and click the "Write" button. Now you can Work on other nefarious schemes during this time.

Linux Command Line

`dd` is already included in many distributions, so you don't need to download anything.

`xz` is already included in many distributions, but if you need to install it, use a command like sudo apt-get install xz-utils (method varies based on flavor of Linux).

Enter `unxz my-odroid-image.img.xz` to extract the .img file (replace "my-odroid-image.img.xz" with the name of the image file downloaded).

Enter `df -h` and make note of mounted devices.

Insert microSD/eMMC card and run `df -h` again. There should be at least one new device in the list. It will look something like "`/dev/mmcblk0p1`" or "`/dev/sdd1`". The last part ("`p1`" or "`1`", respectively) is the partition number. However, to write to the whole card, you need to remove that part from the name ("`/dev/mmcblk0`" or "`/dev/sdd`" from the above example) as the device for the whole card. The card can show up more than once in the output of "df" if there is more than one partition on the card already. Unmount the card with the following:

```
sudo umount /dev/mycard
```

where "`/dev/mycard`" is replaced by the full partition name you found on your system (like "`/dev/mmcblk0p1`" in the example above). you need to run this command for each partition that is mounted on my card.

Remember, "dd" is nicknamed "disk destroyer" for a reason. If you get the following command wrong, this may erase the computer's entire hard drive. (That gives me an idea: Run this command in the opposite configuration as a fail-safe in case my computer falls into the hands of a nemesis!)

```
sudo dd bs=1M if=my-odroid-image.img of=/dev/mycard
```

Replace "my-odroid-image.img" with the path to the image file you downloaded and decompressed.

Replace "`/dev/mycard`" with the full card name you noted in the steps above.

This will take many minutes to complete, so move on to other nefarious schemes while this advances. There will be no output indicating progress. If you want to be notified with an audio alert when "dd" completes, I will enter the following in place of the "dd" command above:

```
sudo dd bs=1M if=my-odroid-image.img of=/dev/mycard && echo -e '\a' || (echo -e '\a'; sleep 1; echo -e '\a')
```

You will hear one beep if "dd" completes successfully, and two beeps if it fails for any reason.

When "dd" finishes, run `sudo sync` to flush the write cache (just in case).

Run `df -h`, use the same method as above to unmount the card partitions, and then remove the card from the computer.

Several hours later, in the wee hours of the morning...

You hear thunderstorms approaching. The hundreds of flashing lights on the equipment in your lab dim momentarily as the power grid attempts to cope with the fluctuations caused by the lightning strikes. You've made changes to the card from which you booted your ODROID. Not wanting your schemes to be foiled by an elementary technological glitch, you decide to make a backup.

One of the simplest ways to do this is with Win32DiskImager on Windows. There are ways to do it on Linux as well (i.e. dd if=/dev/mycard of=/home/user/My-odroid-image.img bs=8M), the storm is approaching quickly. You insert the card into your Windows workstation, start Win32DiskImager, select the drive letter for your card, enter a file name for the backup image, and click "Read". The contents of the card are then read from the card and written safely.

LINUX GAMING ON ODROID

THE NEXT EVOLUTION IN GAMING

Tobias Schaaf
@ schlurf82@googlemail.com

Gaming has always played a major part in my life and is a main reason for my becoming involved with ODROIDs. In this article, I will focus on Linux gaming, because I find Linux to be a much more interesting operating system than Android. Also, most games on Android were made for a touchscreen and rarely support controllers. There are some games for Android that support keyboard or controllers, and run fine on an ODROID as well. They look great on the ODROID and are fun to play, but if you already have an Android device at home, then you already know that an ODROID can play any Android game on a HDTV.

What I find much more interesting is Linux gaming, especially developing games and emulators to run on a system like ODROID in a desktop environment that offers many more options than Android does. For example, I prefer combining different elements into a full operating system package to do exactly what I want, like my ODROID GameStation Turbo image. GST combines a Media Station with a gaming platform as a single package, but in fact it is made of many parts that neatly combine and interact with each other. This level of customization is much more difficult to achieve with an Android OS.

As a side note, Android is based on Linux as well, but still I consider it as a separate operating system, since it acts in many ways different than a normal Linux environment.

First approaches

When the ODROID U2 and X2 were still new, and the Mali 3D drivers were just fresh out of the box, I wondered what we could do with them. I searched the net for OpenGL ES (GLES) games, but it was very hard to find GLES ports of games. Most of the games and programs I found for GLES were only made for Android devices, and it was quite hard to find native GLES ports. Although @mdrjr (one of the lead Hardkernel developers) found a Quake 3 port that he was able to compile against our Mali drivers, it was hard to find something else to run on ODROID that would use GLES anywhere.

Descent - Comparsion between software and hardware rendering

After searching, I found my first game to compile against GLES: Descent 1 and 2 Rebirth. The game performed very well on the ODROID at a steady 100 FPS (frames per second) which was quite awesome, and using GLES made the game look really nice.

Besides Descent, it was really hard to find anything made for Linux using GLES, so I checked what other options I had for Linux gaming on the Odroid.

An easy way to see what was working was to look through the Ubuntu Software Center on the Linaro 12.11 Image (based on Ubuntu 12.04) provided by Hardkernel. There are hundreds of games in the Software Center and many of these are 2D games which worked out of the box. I quickly found old classics on the Internet, such as Dune Legacy which is a Dune 2 Clone with precompiled images for armhf which work without modification.

Seeing these types of games and looking a little bit in the techniques behind these games, I came to a realization: Although these games were made for x86 systems and used mainly CPUs instead of GPU power, these games ran well on the ODROID also. This demonstrates that the ODROID in fact has quite a punch when compared to other ARM boards such as the Open Pandora or Raspberry Pi.

While boards like the Open Pandora or the Raspberry Pi had to build their own ports of many games and emulators (e.g. Amiga Emulator), which need a lot of tweaking and optimization, we were able to use the already existing x86 emulators and just re-

SDL games and ports reference:

http://en.wikipedia.org/wiki/List_of_games_using_SDL
http://libregamewiki.org/SDL_games

compile them for ODROID. The CPU is powerful enough to run those kind of programs and games in software mode.

What makes it work is SDL, which stands for Simple DirectMedia Layer. Lots of programs and games run on SDL and are quite easy to compile for the ODROID platform.

Left: freedroid RPG
Above: Corsix-TH
Far above right: Legend of Edgar
Ab ove right: Open Jazz
Down right: OpenTyrian
Far down right: Jagged Alliance 2 - Stracciatella
Down - PainTown
Down left - OpenXCom

The Magic of SDL

Searching for SDL and 2D helped me to find a lot of games and gaming system emulators. So ports like freedroidRPG, Corsix-TH (a Theme Hospital Clone), Flare, KoboDeluxe, Legend of Edgar, Open Jazz, OpenXCom, OpenTyrian, Zod Engine, PainTown, Jagged Alliance 2 - Stracciatella and ScummVM were possible to run on the ODROID.

While porting a single game is interesting and can be a lot of fun, it's even more interesting to be able to play hundreds of games with just one port: Using an emulator.

Emulators are a good way to get old console games like NES, SNES, SEGA, GBx, Amiga, NeoGeo, or even PlayStation 1 games to run on today's PCs. Again, while other boards have issues with porting some emulators because of the limited CPU power, the ODROID is able to run even emulators like SNES and Amiga in SDL with ease. I could fill this article with thousands of pictures of games of every emulator that is supported on the ODROID.

One of the first multi-system emulator that I used is Mednafen, which allows you to run many emulators with just one program, and was one of my first big breakthrough in emulator ports, since it supports software scalers like hq2x and others.

Playing games on emulators is a very nice experience of playing old games with modern hardware. Games that were once played on small CRT TVs are now able to run with higher resolutions on big TFT TVs while sitting on the couch with a wireless controller.

The power of ODROID combined with SDL makes it all possible.

Technology Advances: OpenGL ES (GLES)

Since the original ports that I did, some time passed and a lot of people worked hard on making the ODROID better. An increasing number of GLES games and programs have been ported to Linux along with the Android version. There are now quite a few games that run under GLES and have great graphics. Games like Hurrican (a Turrican free clone) and Extreme Tux Racer are running in GLES and perform well on the ODROID.

But we were not the only group working on porting GLES games to ARM boards, and so with the help of @pitSeb of the Open Pandora community, who's done some very nice work on porting games that are using the Quake 3 engine, it has been possible to port even more awesome games to ODROID:

We now have RetroArch, a very modular built multi-system emulator, which runs on GLES and uses different cores to emulate different systems. This allows for many consoles to be emulated smoothly on ODROID. Currently, the options are only limited by the Mali binary blobs that were provided by Samsung. @AreaScout, a contributor to the

Above: Jedi Knight 3 Jedi Academy
Above left: Conflict: FreeSpace – The Greater War
Right: FreeSpace 2
Below left: Open Arena
Far above right: UFO - Alien Invasion

ODROID forums, shows that with his Lima driver port of RetroArch, video performance is increases two to three times over the standard Mali drivers.

The options for gaming on the ODROID are quite extensive, with new ports for PSP emulators and NDS. A group has even ported Dolphin (Nintendo Wii and GameCube emulator) to run on ARM boards. Not everything is running smoothly just yet, but with the number of people devoted to ARM development, and with ARM boards becoming faster and faster these days, more will be possible in the future.

Hardkernel offers with each generation of boards the most current "state of the art" hardware. The latest ODROID-XU is even more powerful than the X2 or U2, and emulators that may run slowly on those architectures may perform much better on the XU, and even better on the next generation of Hardkernel computers.

The future is bright

Gaming on Linux was, for a long time, something that people did not take seriously. But with Steam now officially running on Linux, game packages such as the Humble Bundle, as well as more and more games being ported to Linux, it's only a matter of time until these games are also available for ARM powered devices.

Canonical is focusing on the mobile market with a unified design of Linux. The ODROID brings us another step closer to connecting your smartphone to a docking station which will turn it into a powerful desktop PC.

USING ODROIDS IN COMMERCIAL APPLICATIONS OF HIGH PERFORMANCE COMPUTING (HPC)

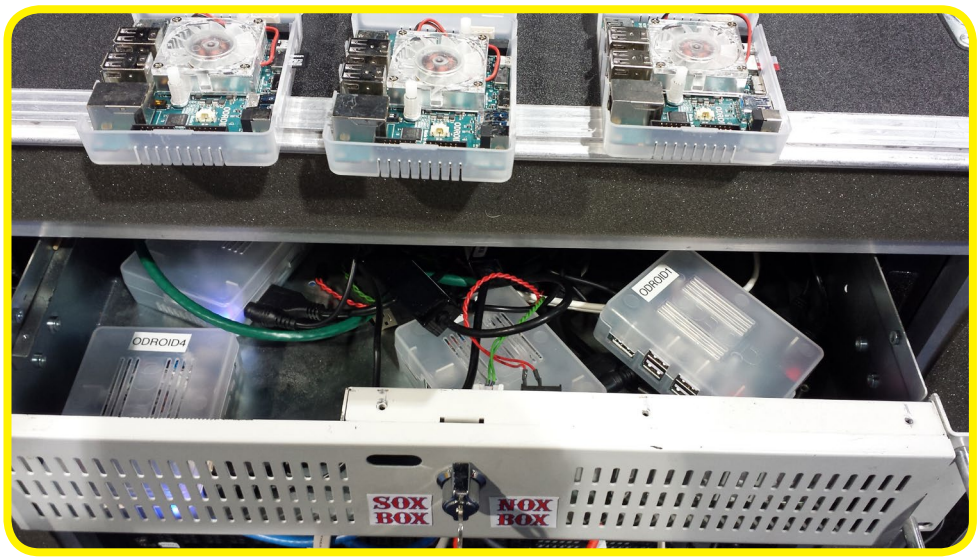
Anthony Skjellum, Cooper Filby, RunTime Computing Solutions, LLC
@ http://www.runtimecomputing.com

Kurt Keville, MIT
@ http://meegs.mit.edu

ODROID technology, based on multicore ARM and General Purpose Graphical Processor Unit Computing (GP-GPU), offers compact microserver technology that is poised to disrupt the world of cloud computing, parallel computing, and embedded systems. In this article, we highlight some of the cool technologies in the present systems, priced at less than US$200 (not including mass memory add-ons).

big.LITTLE Process Scheduling

The ODROID XU+E sports the ARM big.LITTLE architecture which pairs a low power, energy efficient processor with a high power processor; in this case the Cortex-A7 and Cortex-A15. With eight cores, how are processes distributed between them? The overall goal of the big.LITTLE architecture is to reduce power consumption by only using the more powerful A15 cores when necessary, and the kernel process scheduling algorithms reflect this. Based on the data we've gathered using the sensors on the XU+E, we've concluded that the cluster migration process scheduling algorithm is in use. In this scheduling algorithm the A7 processors are used until the processor load reaches a certain level, at which point all the A15 cores power on and take over computation until the load decreases again. Other big.LITTLE scheduling algorithms include In-Kernel Switcher (IKS) -- which pairs each A7 core with a single A15 core for process migration -- and Global Task Scheduling (GTS) which distributes processes among cores based on their processing needs. This means that lower power and high performance are possible in a single architecture. The powerful A15 processor is complemented by a GP-GPU that enables single instruction multiple data acceleration in addition to single-precision vector processing with the A15 NEON floating point extensions.



For many classes of cloud applications, single precision is more than sufficient for floating point computations.

Cluster and Puppet Management

With the ODROID XU+E's and Ubuntu, we were able to successfully setup and configure an HPC cluster using software freely available through apt. For optimal node management, we made use of Puppet, which is a content management system for Linux systems. Using Puppet, we were able to create a baseline configuration for all ODROIDs in our cluster, ensuring a consistent configuration of our nodes. Furthermore, if we ever need to add more software to our baseline configuration, we just need to update the configuration on our puppet server and the changes will sync to the rest of the cluster. The flexibility of Puppet allows us to do even non-trivial tasks such as configuring local users, autofs and LDAP, all while making it easy for us to add new nodes.

Cloud Infrastructure Applications

Cloud computing based on many low power ODROID XU+E's offers an interesting alternative to a fewer number of highly virtualized x86-64 processors. By providing many more smaller systems, together with gigabit networking enabled by the XU+E's high performance USB 3.0, cloud architectures that emphasize assurance at the network layer are possible, as are "more traditional" architectures that paravirtualize and achieve trust through the host OS layer. Because the systems are relatively inexpensive, greater control of physical hardware within a cloud infrastructure becomes possible. This is analogous to some of the projects in the wild currently, such as HP Moonshot, but is real today, without proprietary lock. Anyone can build an XU+E cloud. Furthermore, the I/O capability of the eMMC modules offer I/O performance exceeding 100Mbyte/s and high IOP counts in capacities currently up to 64 gigabytes. This will support acceleration of a number of classes of workloads.

Supporting High Performance Math Libraries

Both open and commercial mathematical libraries have or will soon have support for ODROID. ATLAS, for instance, (http://sourceforge.net/projects/math-atlas) is in active development for the ARM family and has already shown benefits to many applications that utilize BLAS routines.

Furthermore, commercial libraries for signal and image processing are in development, particularly based on the VSIPL standard (http://www.omg.org/hot-topics/vsipl.htm). For instance, RunTime Computing Solutions (www.runtimecomputing.com) will soon release VSIPL for the multicore ARM. This enables transition from older platforms such as the PowerPC Altivec processor family directly to ARM with performance portability, and achieving performance higher than open source libraries such as FFTW.

ODROID Caveats

Unfortunately, no bleeding edge system is without certain difficulties, and the ODROID XU+E is no exception. There were two difficulties that we had to overcome when first setting up our cluster with NFS and LDAP. We found that the kernel used in the Xubuntu 13.10 image did not actually support the NFS kernel daemon. This was easily overcome by building the freely available UNFS3 and running it on the head node. Secondly, we ran into issues when trying to configure the nodes to authenticate against the LDAP server on the head node when using nscd and ldap-auth-client. We found that the solution to this issue was to symbolically link the /lib/arm-linux-gnueabihf/nss_ldap.so file to /lib/libnss_ldap.so.2. The good news is that the Linux support is evolving rapidly, and we have seen great improvements in performance and functionality with each release.

Conclusions

The HPC opportunities available on ODROID are legion. We can now perform traditional HPC activities like netbooting, NFS mounting, multipath network routing, cpufreq control associated with load (which can then be attached to a queueing system like slurm), and compile and link modern supercomputing applications against enterprise libraries like MPI and OpenMP.

And, by moving to ARM now for cluster and HPC, we can rapidly track the emerging roadmap of ODROID microservers, which are rapidly tracking the release of better and better ARM processors.

REAL-TIME MOBILE EYE TRACKING ON THE ODROID DEVELOPMENT PLATFORM

Christopher D. McMurrough
@murrman@sbcglobal.net

In this article, we provide an introduction to eye tracking using a low-cost, open source headset and the ODROID-U2 development platform. Our goal is to demonstrate the potential of the processor as a platform for mobile, real-time video oculography. The experimental setup can be duplicated relatively inexpensively due to the availability of low-cost hardware components and open source software.

INTRODUCTION

Video-based eye tracking, or video oculography, is an emerging technology that has the potential to change the way in which people interact with intelligent systems. The utility of eye tracking has already been demonstrated for users with severe physical disabilities, such as paralysis, ALS, and neurological conditions, by providing hands-free typing functionality. In these systems, human-computer interaction is provided by tracking the user's point-of-gaze (PoG) on a computer display rather than relying on a standard keyboard and mouse or speech recognition.

Many different types of video oculography devices exist both commercially and in research, but the general operating principle remains the same. First, a camera (usually infrared) observes the user's eye and estimates the location of the pupil. Next, a calibration process pairs pupil positions to known monitor coordinates while the user looks at target points on the display. Finally, an estimated PoG is interpolated using the calibration data points.

While eye tracking is an exciting new technology, commercial devices tend to be quite expensive. There are also in general many open problems that currently prevent the technology from being more widely used. These limitations, however, in no way prevent the hobbyist or aspiring researcher from performing their own experiments! In this article, we provide insights on how to get started with mobile eye tracking on the ODROID using open source hardware and software.

User wearing the EyeWriter headset

HARDWARE

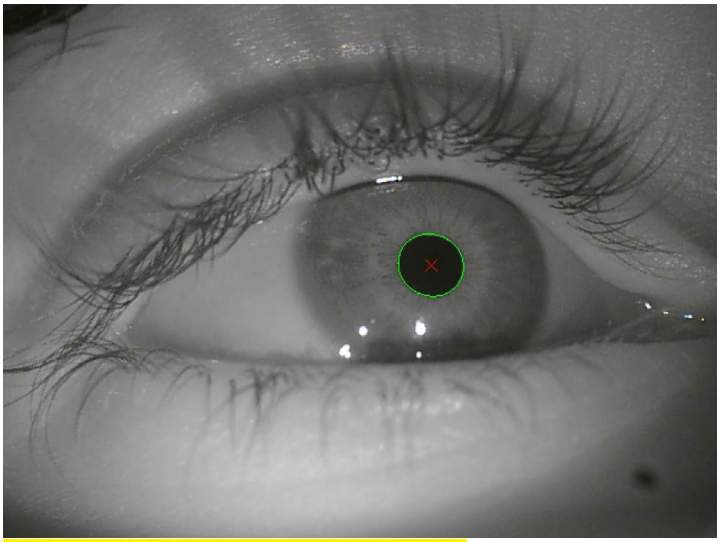
As mentioned previously, commercially available eye trackers tend to be cost-prohibitive for hobbyists and experimenters. Luckily, there are a few designs that can be made by hacking common consumer electronics. In this article, we use an EyeWriter headset [1]. This headset can be created by attaching a modified PlayStation Eye Camera to a set of cheap sunglasses using a flexible aluminum wire arm. Surprisingly, the video quality and frame rate provided by the properly modified camera meets or exceeds that of many commercially available head-mounted eye trackers. Because of the ease of construction and favorable performance characteristics of the EyeWriter, we will utilize this device for our experimental setup. An image of our test volunteer wearing the headset is shown above.

Once we have acquired our oculography headset, we need to process the eye video in order to locate the pupil. We perform this processing on an ODROID-U2 development board. The U2 was chosen due to its relatively small size, low cost, and image processing potential. We will also take advantage of several open source software projects running on the Ubuntu OS, which is supported by the ODROID.

SOFTWARE

Our system relies on image processing techniques to track the pupil from video frames provided by the USB camera. Specifically, we will use the open source Robust Pupil Tracker [2], an OpenCV application written in C++. This application attempts to fit an ellipse around the pupil in a video oculography frame. The center of this ellipse then provides an accurate estimation of the pupil center in pixel coordinates, which we can use in our own applications. An example of a processed video frame from the experimental headset is provided below.

In order to run the pupil tracker, we need to set up the Ubuntu OS on our ODROID-U2. We also need to install the OpenCV, Boost, and Threaded Building Blocks libraries on our system, which are dependencies of the pupil tracking application. This process requires some compilation from source and can be a bit tedious. Luckily, we can skip these steps by downloading the Ubuntu 12.11 Robotics Edition [3] OS image from the ODROID forums. This image contains the libraries required by this project, as well as other useful robotics and perception tools such as Robotic Operating System (ROS), OpenNI, Point Cloud Library (PCL), etc. For our setup, the OS image was downloaded and installed to an 8 GB SD card. Upon booting the ODROID-U2, we are able to log in with the default credentials to run our experiments.



Video oculography image with pupil tracking

EXPERIMENTAL RESULTS

Pupil tracking in video oculography images, in general, can be a computationally expensive process. Several image processing steps must be performed sequentially in order to achieve accurate results. In order to evaluate the performance of the ODROID, we ran the Robust Pupil Tracker on real-time oculography frames provided by the EyeWriter headset on 3 different platforms: A high end quad-core desktop PC, a single core Intel Atom Netbook, and an ODROID-U2.

We have set the raw video frames provided by the EyeWriter USB camera to arrive at a rate of 30 FPS (frames per second) at a resolution of 640x480 pixels (the camera is capable of higher performance, but these settings work well for our application). Video frames were acquired and processed in real-time for 30 seconds on each platform. The actual processing frame rates for each platform are shown in Table 1.

As demonstrated by the experimental results, the ODROID-U2 provides a performance increase over a standard netbook. This is quite impressive, given the physically smaller size, lower power requirements, and lower monetary cost relative to the netbook. While a frame rate of 11.8 fps may not be good enough for tracking saccade pupil motion (rapid eye motion during visual scanning), this is still a good rate for most real-time applications, particularly detection of gaze fixation for communication and control.

Comparison of processing frame rates

Platform	Processed Framerate (FPS)
Desktop PC	30.0 FPS
Netbook	9.8 FPS
ODROID-U2	11.8 FPS

CONCLUSION

Eye tracking is an exciting emerging technology that, to a great extent, is being driven by the open source community. The topic itself has many open problems, but experimenters and hobbyists can get started by taking advantage of various publicly available hardware and software designs. Video oculography has already made a difference in the lives of many people with severe physical disabilities, and with further development, the field can provide a positive impact for even more users.

REFERENCES

[1] Z. Lieberman, C. Sugrue, T. Watson, J. Powderly, E. Roth, and T. Quan, "The EyeWriter," EyeWriter Initiative, 2009. [Online]. Available: http://www.eyewriter.org.

[2] L. Świrski, A. Bulling, and N. Dodgson, "Robust real-time pupil tracking in highly off-axis images," in Proceedings of the Symposium on Eye Tracking Research and Applications - ETRA '12, 2012, p. 173.

[3] C. McMurrough, "Ubuntu 12.11 Robotics Edition v2 (ROS+OpenCV+OpenNI+PCL) U2," ODROID Forum, 2013. [Online]. Available: http://forum.odroid.com/viewtopic.php?f=8&t=2096

RUNNING LINUX PROGRAMS UNDER ANDROID

A GUIDE TO LINUX CHROOT

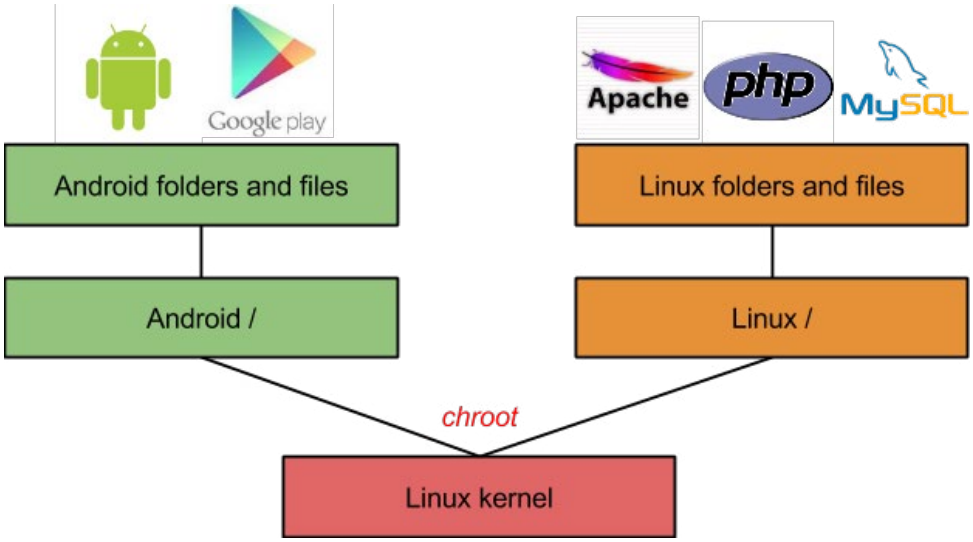
Marian Mihailescu
@ mihailescu2m@gmail.com

Android is Google's user-friendly operating system that is available for many mobile and compact devices worldwide, including the ODROID. Like Debian and Ubuntu, Android is based on the Linux kernel. This can be easily verified by going into the Terminal application from your Android device and running:

```
root@android:/ # uname -a
Linux localhost 3.0.51-g4732a09
#3 SMP PREEMPT Sun Aug 4 07:26:36
CST 2013 armv7l GNU/Linux
```

Unlike Linux, however, Android has its own windowing system that replaces X.org, separate libraries, and a somewhat different general layout of the root filesystem. The root filesystem or the root directory is the one from which the system boots up, commonly referred to as / (slash), the location where all the other filesystems are mounted, and thus where all files reside. Each application knows where the root directory is by asking the kernel (using system calls). chroot is a method of instructing the kernel (using the chroot system call) to change the apparent root directory for the current application and its children.

Since both Linux and Android use the same kernel, it is possible then to have two different root filesystems (one for Linux and one for Android), and, by using chroot, set a different one depending on the application you are running - an Android application would use the Android root filesystem, while a Linux application would use the Linux root filesystem. Since there is only one kernel running, both filesystems would share the same resources. Hence, there is no performance penalty or overhead when using chroot, since each application accesses the same kernel directly.



Applications from different Operating Systems running on the same kernel

When working with chroot, there are a few issues you have to overcome:

First, the system that you run in chroot (client) will start from an already "booted" state, so all initializations and startup scripts will not be executed. Of course, this is actually a good thing, since the original root filesystem that booted first (host) probably did similar things already. However, whatever programs that would have otherwise been started at boot in the client would need to be started manually when using chroot.

Second, the Linux kernel uses several "virtual" filesystems to manage resources, ranging from hardware components to running applications. On the host filesystem, these virtual filesystems are created and mounted at boot, so for the clients you need to mount them yourself (since resources are shared, you need to mount the same filesystems created on the host). Lastly, you need to have the client root filesystems already mounted on the host's running filesystem. It is the user's choice where the client root filesystems are physically located on a partition on the same drive as the host filesystem, on a different drive, or even on an image file that acts as another drive.

For the purpose of this article, the host filesystem is Android, and the guest filesystem is Linux (Ubuntu) running from an image. Using an image has several benefits, such as portability (you just need to copy one file to move the entire filesystem), ease of use (most distributions provide images for their root filesystems), and better compatibility with Android (Android requires a certain partition layout, and it's not recommended to change it). As a requirement, you need a Terminal application on Android to access the command line, and a set of commands available with Busybox. Busybox is an application available in Android that provides several stripped-down Unix tools in one executable.

Assuming you have an image with the client (Ubuntu) root filesystem, you first need to mount it under Android, in this case in the /data/local/ubuntu folder:

```
root@android:/sdcard/ubuntu # busybox mknod /dev/block/loop127 b 7 127
root@android:/sdcard/ubuntu # busybox losetup /dev/block/loop127 ubuntu.
img
root@android:/sdcard/ubuntu # mkdir /data/local/ubuntu
root@android:/sdcard/ubuntu # busybox mount -t ext2 /dev/block/loop127 /
data/local/ubuntu/
```

Navigating to the folder where the image was mounted, the Linux root filesystem is now visible:

```
root@android:/sdcard/ubuntu # ls -la /data/local/ubuntu
drwxr-xr-x root     root              2012-03-29 09:13 bin
drwxr-xr-x root     root              2008-08-05 16:38 boot
drwxr-xr-x root     root              2012-03-29 09:19 dev
drwxr-xr-x root     root              2012-06-07 14:19 etc
drwxr-xr-x root     root              2009-01-10 18:58 home
drwxr-xr-x root     root              2012-03-29 09:13 lib
drwx------ root     root              2008-11-08 03:56 lost+found
drwxr-xr-x root     root              2008-11-08 04:50 media
drwxr-xr-x root     root              2008-08-05 16:38 mnt
drwxr-xr-x root     root              2008-11-08 04:50 opt
drwxr-xr-x root     root              2008-08-05 16:38 proc
drwxr-xr-x root     root              2012-06-30 14:45 root
drwxr-xr-x root     root              2012-03-29 09:13 sbin
drwxr-xr-x root     root              2008-11-08 04:50 srv
drwxr-xr-x root     root              2008-08-12 14:26 sys
drwxrwxrwt root     root              2012-06-07 14:19 tmp
drwxr-xr-x root     root              2009-01-08 21:35 usr
drwxr-xr-x root     root              2008-11-08 04:50 var
```

Next, you need to mount the existing virtual filesystems created by the Linux kernel in Android, under the corresponding locations in the Linux filesystem:

```
root@android:/sdcard/ubuntu #
busybox mount -t devpts devpts
/data/local/ubuntu/dev/pts

root@android:/sdcard/ubuntu #
busybox mount -t proc proc /
data/local/ubuntu/proc

root@android:/sdcard/ubuntu #
busybox mount -t sysfs sysfs
/data/local/ubuntu/sys
```

You can also mount other filesystems from Android if you want to have them available in Linux, for example the sdcard:

```
root@android:/sdcard/ubuntu
# mkdir /data/local/ubuntu/
media/sdcard

root@android:/sdcard/ubuntu #
busybox mount -o bind /sdcard
/data/local/ubuntu/media/sd-
card
```

Make sure the network is enabled and forwarded for the Linux filesystem:

```
root@android:/sdcard/ubuntu
# busybox sysctl -w net.ipv4.
ip_forward=1
```

Now, you just need to start a Linux command prompt that thinks the root filesystem is Linux:

```
root@android:/sdcard/ubuntu
# busybox chroot /data/local/
ubuntu/ /bin/bash

root@localhost:/#

root@localhost:/root# cat /
etc/issue

Ubuntu 12.04 \n \l
```

Start running your Linux programs now, install new programs, etc. It's a good idea at this point to install the ssh daemon, if it's not already installed. With ssh started, you can access your Linux system without going through all these steps again.

Since this root filesystem was not actually booted, there is no initialization done. There are some daemons and several environment variables that you would want (like ssh for example).

```
root@localhost:/# ex-
port PATH=/bin:/sbin:/
usr/bin:/usr/sbin:/
usr/local/sbin

root@localhost:/# ex-
port TERM=linux

root@localhost:/# ex-
port HOME=/root

root@localhost:/# ser-
vice ssh start
```

Set up the host system to have Linux available after each Android boot by collecting the above commands in a script and run the script when Android boots (there are several applications in the Android to run scripts on boot). Create a script in the Linux image with all the initializations to be done on boot, (environment variables and daemons), and then be sure you run that script under chroot (e.g. /root/init.sh):

```
root@android:/sdcard/ubuntu
# busybox chroot /data/local/
ubuntu/ /bin/bash /root/init.
sh
root@localhost:/#
```

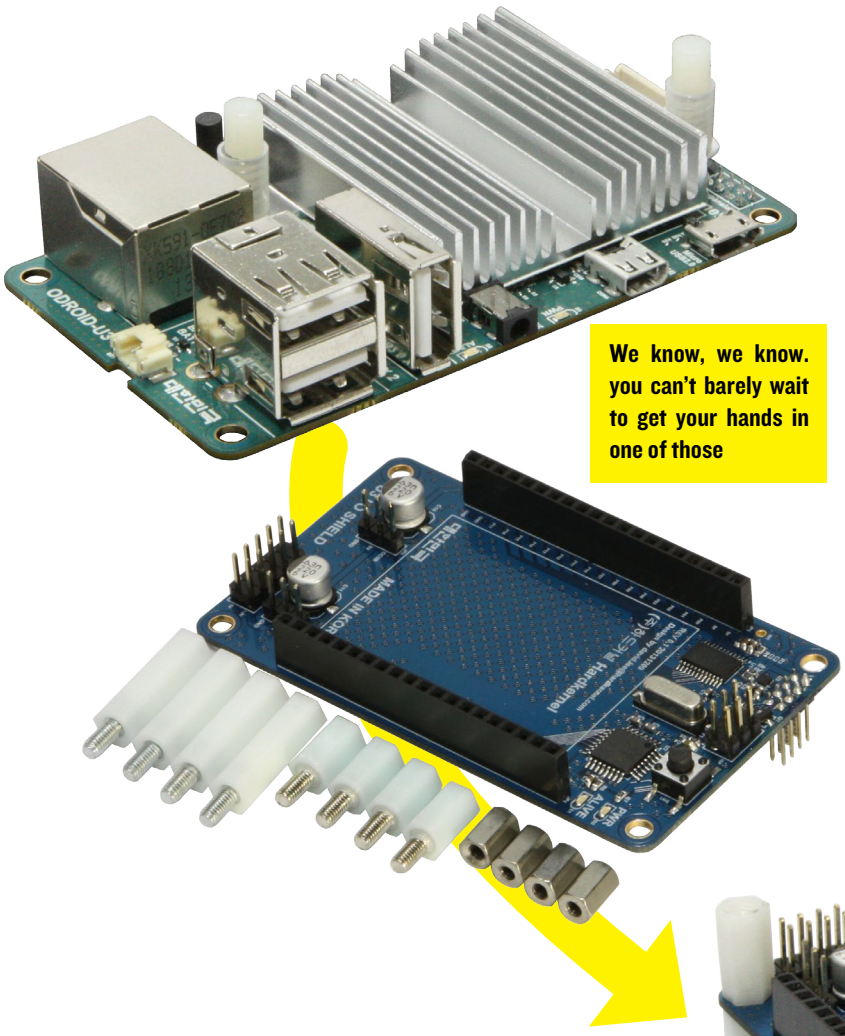
The follow are useful to add, and from there the possibilities are endless!

```
# environment variables
export PATH=/bin:/sbin:/usr/
bin:/usr/sbin:/usr/local/
sbin
export TERM=linux
export HOME=/root

# fix some issues with up-
start
dpkg-divert --local --rename
--add /sbin/initctl > /dev/
null 2>&1
ln -s /bin/true /sbin/initctl
> /dev/null 2>&1

# create /etc/mtab file with
the mounted filesystems
gep -Ev "rootfs|tmpfs|cgroup|
mmcblk|usbfs|asec|storage" /
proc/mounts | sort -r | uniq
> /etc/mtab

# start ssh daemon
service ssh start
# start other daemons…
```

IO PORTS ON ODROID-U3

GET STARTED WITH TINKERING

Justin Lee
@justin.lee@hardkernel.com

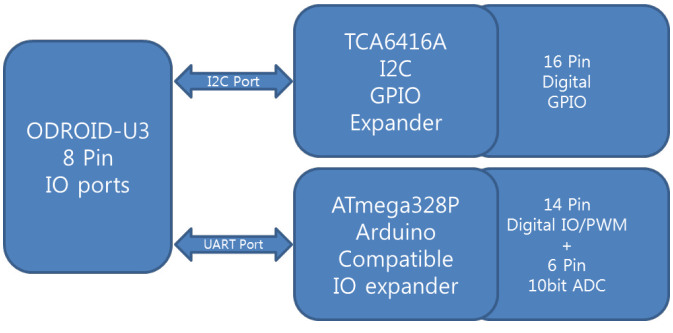


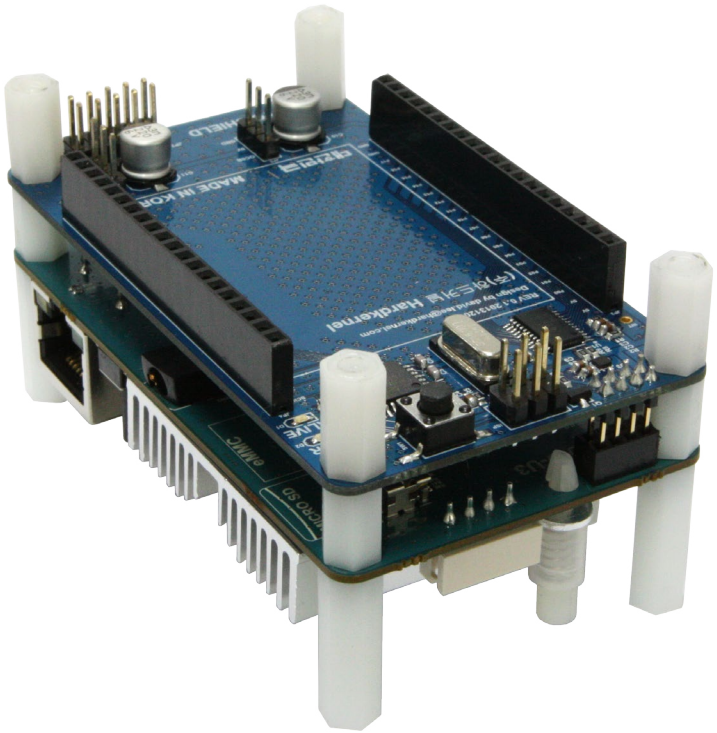
We know, we know. you can't barely wait to get your hands in one of those

You may want to connect a few servo motors, LEDs and switches to the ODROID-U3. But there is only one 8-pin connector, so how do you get more? The power rails are 5V, 1.8V and Ground, leaving only 5 ports for use. Even more confusing is that they use a 1.8V interface, but most applicationsn require a 5V interface. How does an aspiring hardware tinkerer resolve these issues?

Don't worry, there is a good solution available: the IO Shield board. An IO Shield board can be plugged into the ODROID-U3 using a tiny 8 pin IO connector. The shape and dimension are exactly same as ODROID-U3, and it comes with 12pcs of PCB spacers for easier and solid stacking assembly.

The IO shield has 2 blocks. "I2C IO expansion block" and "Arduino Compatible block"

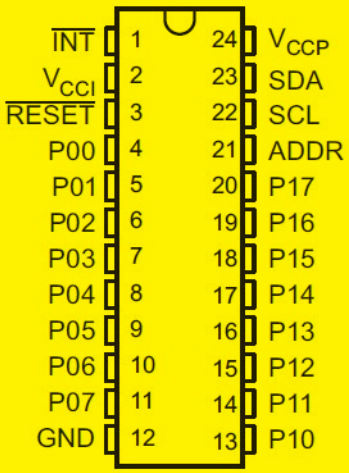




I2C IO expander: 16 x GPIO

TI's TCA6416A I2C to Parallel Port Expander is used in this block.

The major benefit of this device is its wide VCC range. It can operate from 1.65 V to 5.5 V on the P-port side and on the SDA/SCL side separately. It allows bidirectional voltage-level translation GPIO Expansion between 1.8V SCL/SDA(VCCI) and 5Volt Port(VCCP).



16 Ports are mapped to gpio#289~#304, which can be read from the command line. Here is an example to access a GPIO. Note that you need to load the driver first with a modprobe command.

```
# modprobe gpio-pca953x
# echo tca6416 0x20 > /sys/
devices/platform/i2c-gpio.4/
i2c-4/new_device
# echo 289 > /sys/class/gpio/
export
# cd /sys/class/gpio/gpio289
/sys/class/gpio/gpio289#
echo "in" > /sys/class/gpio/
gpio289/direction
/sys/class/gpio/gpio289# cat
direction

in

/sys/class/gpio/gpio289# cat
value

1
```

You can use any generic C/C++ or Python libraries to access the 16 GPIOs in your program.

Arduino Uno compatible IO

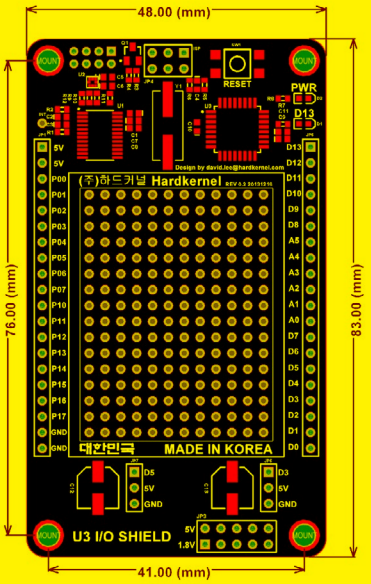
Atmel's ATMEGA328P is used as a slave CPU to expand the IO port. You can use the Arduino IDE on ODROID-U3 to build and upload your sketches to this Arduino-compatible block, without additional/external cable connections. Just choose the "/dev/ttyACM99" in the Menu >> Tools >> Serial Port configuration. ATMEGA328P is a slave device on UART bus and ODROID is a master.

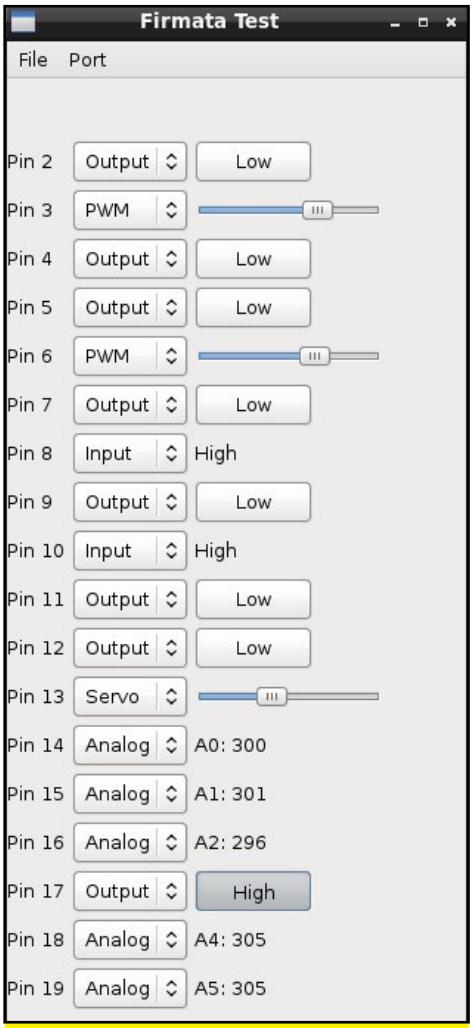
Like the Arduino Uno, it has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs which provides 10 bits of resolution. They operate at 5 volts range.

Note that the ATmega328P in the IO Shield board is shipped with the Arduino IDE compatible boot loader optiboot-v5.0a and StandardFirmata Ver2.3 firmware.

If you need to quickly test your hardware, this stand-alone program can access all pins. Here is the source code of this test suite: http://www.pjrc.com/teensy/firmata_test/firmata_test_OSL.tgz.

Here is a PCB CAD drawing for your reference. The PCB Gerber files will be released soon for your own DIY shield build.





At Hardkernel, we tested the Firmata (`http://firmata.org`) to make an interactive interface between ODROID-U3 and ATmega328P. The Firmata is a generic protocol for communicating with microcontrollers from software on a host computer.

The IO Shield board is designed as versatile, general-purpose data acquisition and control module and provides a direct connection to ODROID-U3 IO connector. We added lots of basic but essential goodies. First up, there's a big prototyping area, so you can wire up DIP chips, sensors, and the like. Along the edges of the proto area, all the GPIO/ADC/PWM and power pins are broken out to 0.1" stips so you can easily connect to them. There are also two 3-pin headers for small servo motors connection. Finally, we added a 10-pin connector of I2C/GPIO for further expansion.

It's time to make something tinkering and prototyping with your powerful 1.7Ghz ARM quad-core tiny computer.

ON THE PERFORMANCE OF SD CARDS

KNOW MORE ABOUT YOUR STORAGE OPTIONS

Jussi Opas
@jussi0@welho.com

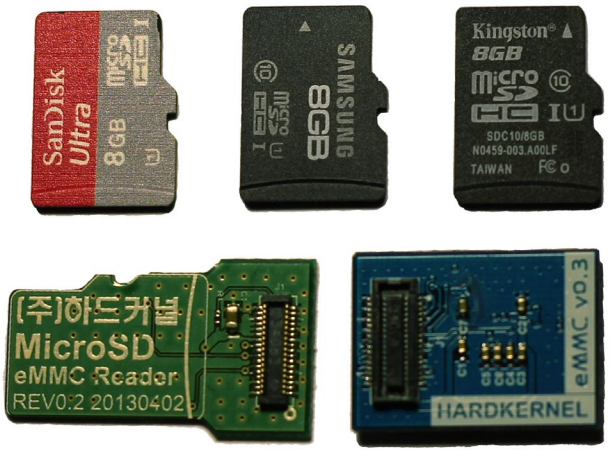
Currently the highest-end micro SD cards available are classified as 10 class UHS-I. By specification, the minimum data read speed of a UHS-I classified card is 10 MB/s. The maximum data read speed is not defined, so each card may exceed considerably the minimum limit. The maximum speed may be advertised to be 10, 20, 40 or even 70 MB/s. In published articles, read/write tests are made with latest SD card readers, which often are not available for ordinary home users. How does this advertised performance manifest itself in use with an ODROID board and a casual user's host computer? In this article, we present an SD card user's experience with self directed tests.

Our tested cards:

SanDisk Ultra 8GB:
As shipped together with ODROID board.
Samsung Pro 8GB:
Which is aimed for video storage for digital cameras and advertised to write at 20 MB/s and read at 70 MB/s.
Kingston 8GB:
Another UHS-I level SD card.
eMMC card 64 GB:
ODROID on board storage. On a host computer it can be used similarly to micro sized SD cards with an adapter that is shipped with the board.

We created reading tests using the hdparm command of Linux. It can be invoked, for instance, as sudo hdparm -t /dev/mmcblk0p1. The SD card read test was made with a 4-year-old laptop (Celeron x86), a 2-year-old laptop (i5 x86) and a new ODROID XU (Exynos ARM) board. The results are shown in the following table:

READ MB/S	2-YEAR-OLD LAPTOP	4-YEAR-OLD LAPTOP	XU
SanDisk Ultra	13.8	19.4	17.6
Samsung Pro	16.0	16.6 - 18.0	16.9 - 18.1
Kingston	11.8 - 14.7	15.7 - 16.8	15.4 - 16.5



All UHS cards read at about the same performance level. Repeated tests may give slightly different results, but the variation is only about 1 MB/s. The 2-year-old laptop reads cards 2-3 MB/s faster than the 4-year-old laptop. The XU reading speed is at about same level with the 2-year-old laptop. None of the read tests come close to the 70 MB/s that was advertised in the case of one of the cards. Therefore, to use the full speed of the fastest SD card, one should either purchase a faster card reader or use another test method.

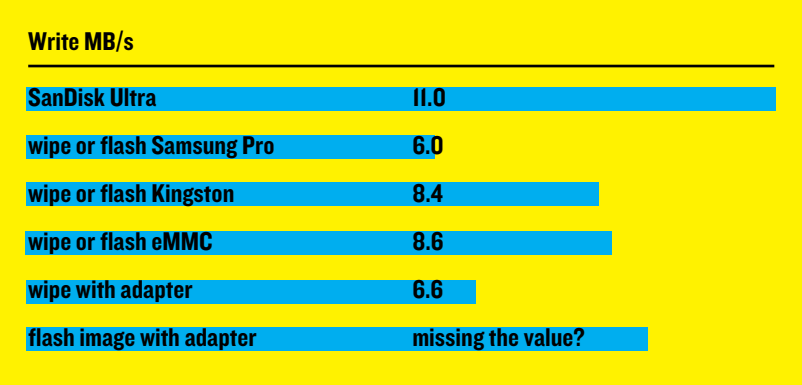
To bring the SD card reading speed in line with other drives and eMMC cards, we show below the hdparm read test results on a couple of hard drives and one SSD installation via an eSATA connection. The results are shown below in decreasing order:

Reading is fastest from an SSD installed on a laptop (500MB/s), and 268MB/s

READ MB/S	
SSD on 2-year-old laptop	500
SSD via eSATA on 2-year-old laptop	268
eMMC on XU	90 - 117
HD on 4-year-old laptop	63
HD on 7-year-old desktop	30
SD on XU	17.6

via eSATA in repeated runs. The result of eMMC read performance varies in repeated runs and between different file systems, whether formatted as FAT or EXT4. The fastest result was 117 MB/s. The hard disk of a 4-year-old laptop can do only half of the eMMC read speed at 63MB/s. Still lower performing is the hard disk of a 7-year-old desktop computer. An SD card achieves the same read speed on an XU board as an older hard drive.

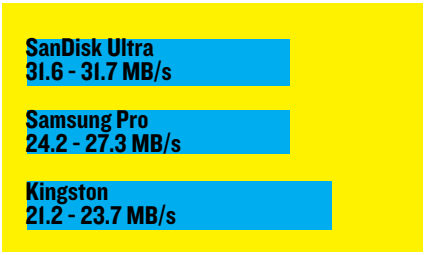
Writing was tested with the dd command. The command yields excellent testing results, because it gives as output the elapsed time and actual writing speed. In practice, dd is used to wipe a card and then to flash an OS image into it (see the OS installation article in this issue for more details). The write results on a 4-year-old laptop are shown below:



Both the 2- and 4-year-old laptops write at about the same speed, indicating that write speed is probably not as dependent on the host hardware. The SanDisk Ultra is the fastest card to wipe and flash with both operations being done at the same speed. It is surprising that the writing speed of the Samsung Pro card is just 6 MB/s, far from the promised capability of 20 MB/s. Kingston performs steadily at 8.4 MB/s. Here we see that wiping an eMMC card is performed at 8.6 MB/s while flashing is done at 6.6 MB/s. To wipe and flash faster with an eMMC card, one should run Linux on an SD card and have the eMMC card installed on the XU board at the same time, but this configuration was not attemped in this testing.

When compared against other similar cards, the SD card shipped with the ODROID board performs equal or better than the other tested cards in this test. The eMMC card performs better than mechanical rotating disks since the read speed reaches approximately half of the speed of an expensive SDD that has been mounted via eSATA.

As of December, 2013, upgrades to the ODROID-XU kernel have greatly improved read speeds. The ODROID-XU with kernel 3.4.70 or newer yields the following benchmark results:



PLAYING YOUTUBE CONTENT ON LINUX

Jussi Opas
@ insert email here

One issue that puzzles computer (and especially Linux) users is why some videos can't be played while others can. There may be situations when a video does not play at all, or one sees only video but hears no audio; or the video is not shown while the audio is being played. The rendering of a video is actually quite a complicated process; to transfer content effectively, there must be an efficient packing and unpacking method, appropriate video codecs, as well as an audio codec. Then, the hardware does its job through the HDMI connection or headphone jack audio port, since both are available on the ODROID-XU board. All these parts need to work together in order to produce an enjoyable video experience.

Codecs may be licensed, or they may be open-source software. Especially in the developing ARM world, an open codec or player may be unavailable for Linux. The most obvious is the absence of the Adobe Flash player, which was discontinued in 2012. If we can live without Flash, this is also an advantage - Flash advertisements cannot play while browsing. But what about video and audio content from YouTube or some other content source? If a user can not solve his/her playing problem, he/she may think that the problem is due to defective hardware. There is a simple and readily available method of solving playback problems, as outlined below.

If there is problem with playing YouTube videos, then one can try playing it in Firefox instead of Chromium, or vice versa. If neither of these works, then one can switch to the built-in HTML5 player in either browser. While viewing the problematic video, right-click and select the lowest menu item titled "About HTML5", and a page with the address http://www.youtube.com/html5 will be opened.

For instance, the current browser versions support WebM video with VP8 audio codec. Press the "Request the HTML5 player" button, reload your video, and there is good possibility that the selected video will work as expected.

Users can see the video format and audio codec of the content by right-clicking on top of the video and selecting the "Stats for nerds" menu item. If the selected video is Mime Type: video/webm; codec="vp8.0, vorbis", it will render flawlessly with Firefox on an ODROID_XU running Xubuntu 13.10.

Happy video watching and music playing to everybody! You can try this right now with your ODROID board.

ODROID-U3 VS ODROID-U2

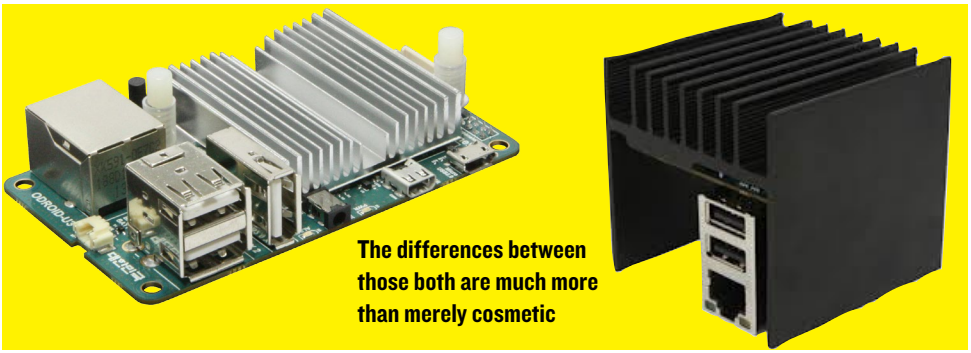
THE NEXT-GENERATION U3 IMPROVES UPON THE POPULAR U2

Justin Lee
@justin.lee@hardkernel.com

We launched the ODROID-U2 board computer in December 2012. It is already a year old, but the U2 is still one of the highest class products in the ARM PC world. The latest Ubuntu 13.10 and Android 4.2.2 both work great on the evolving Linux Kernel 3.8, and many forum members are actively developing software for the ODROID family of computers. So, we decided to improve the hardware of the U2, and the ODROID-U3 was born!

Please note that the ODROID-U3 is 100% software compatible with the ODROID-U2.

There are 7 key differences between the U3 and the U2.



The differences between those both are much more than merely cosmetic

Price drop: $89 to $59

Yes, it is the most important difference! The 34% price drop was made possible with the assistance of many suppliers (including Samsung) as well as our hardware engineers. The normal price is $65, but you can receive the special $59 discount price by registering on our forums at http://forum.odroid.com.

USB Host port

ODROID-U2 has only 2 USB host ports, which was not enough for normal desktop use. For example, if you use a keyboard and a mouse, there is no room for a WiFi adapter. So, we decided to add another USB port for a total of 3 USB host ports. This new feature significantly reduces the need for an externally powered USB hub.

Another improvement was adding a smarter load switch which resolved current leakage to the USB devices when the system is powered down, reducing potential damage to the devices.

Power on/off button

We added a hardware button to power on and power off the ODROID-U3, even though it already has the auto power-on feature. It's no longer necessary to plug and unplug the DC plug. To shut down the system, just press the button and you will see a power-off dialog box on the monitor. To restart, just press the button again and the ODROID-U3 will turn on.

IO ports

ODROID-U2 had no exposed IO ports, so we used the USB interface to add sensors, motors, LEDs, switches, etc. A tiny 8-pin connector on ODROID-U3 contains the UART, IRQ, I2C, GPIO and power.

For those who are considering using an ODROID-U3 for a robotics application, our full featured U3-IO-Shield expansion board has 36 IO ports including 24 GPIOs, 6 PWMs and 6 ADCs.

Volume (or Dimension)

Many people like the tiny size of U2. But, some people were concerned about the height, and we agreed that the U2 could be shorter. So, we slightly increased the PCB footprint, and reduced the height with a space improvement of about 50%, which means that two U3's can fit into the space of a single U2.

The volume of the ODROID-U2 is 198,360mm3 (58 x 60 x 57 mm with heat sink), and the volume of the ODROID-U3 is 87,648mm3 (83 x 48 x 22 mm with heat sink).

Real time clock (RTC)

By simply adding a coin-sized backup battery, you will be able to keep the clock and calendar accurate when the power goes out, especially for stand-alone projects without a network connection.

Input power protection circuit.

Reverse-voltage, Over-voltage, Under-voltage and ESD protection are all implemented on the power input circuitry with a special IC. It provides overvoltage protection for positive and negative voltages, up to 28 V and down to −28 V.

During overvoltage conditions, the output of the IC remains disabled if the input voltage exceeds 6 V. This gives the ODROID-U3 better endurance against accidental power supply mis-usage such as reverse polarity or high voltage input.

ODROID-U3 VS RASPBERRY PI

NEED MORE BANG FOR YOUR BUCK? U3 WILL GET YOU THERE!

Mauro Ribeiro
@ mdrjr@gmail.com

Justin Lee
@ justin.lee@hardkernel.com

Both are Linux-friendly, cost-effective ARM single-board computers for various applications and purposes.

Although the ODROID-U3 is a cost-effective ARM single-board computer, it sports a quad-core Samsung ARM SoC. The specifications on the ORDROID-U3 include a Samsung Exynos 4412 quad-core ARM Cortex-A9 1.7GHz SoC, Mali-400 MP with 4 GPU Cores, three USB 2.0 host ports, one 10/100MB Ethernet port, 1080p video output via micro-HDMI, 2GB of LP-DDR2 system memory, eMMC and micro-SD for storage, and supports Debian Wheezy, Ubuntu 13.10 Linux and Android Jellybean Operating Systems.

Hardware comparison

Compared to the Raspberry Pi (aka RPi), the number of CPU/GPU cores is 4 times higher and the operating clock frequency of the CPU is about 2.4 times faster. In addition, the RAM size is also 4 times larger and the RAM access frequency is 1.6 times faster. For the overall footprint of the PCB, the RPi is 1.2 times larger and the weight is very similar if we consider the heat sink. You can also use the hardware RTC (real-time clock) with a coin battery on the ODROID-U3.

The RPi uses a Broadcom SoC (System on a Chip) that's an ARM v6 clocked at 700Mhz (it can be overclocked up to 1Ghz; however the tested RPi didn't work beyond 800Mhz). By comparison, the U3 uses a newer ARM v7 processor -- the Samsung Exynos 4412. This is the same chip you'll find in powerful smartphones like the Galaxy Note 2 and the Galaxy S3 phones. It has 4 Cortex-A9 cores run-

Yeah yeah, the RPi isn't even correctly clipped. That's how much it sucks.

ODROID-U3	Raspberry Pi (Model-B/512MB)
CPU	
SAMSUNG EXYNOS-4412 Prime 4 x ARM Cortex-A9 @1.7Ghz ARMv7 Architecture	BROADCOM BCM2835 1 x ARM11 @700Mhz ARMv6 Architecture
GPU	
4 x ARM Mali400 @400Mhz	1 x VideoCore IV @ 250 MHz
RAM	
2GB LP-DDR2 @440Mhz	512MB SDRAM @400Mhz
USB 2.0 Host + device	
3 Ports • 1 Port for Linux USB Gadget driver	2 Ports • NO
ETHERNET	
10/100 Mbit/s	10/100 Mbit/s
VIDEO OUT	
HDMI (480p/720p/1080p)	HDMI / Composite RCA
AUDIO OUT	
3.5mm Jack / HDMI	3.5mm Jack / HDMI
REAL TIME CLOCK	
YES (Backup battery connection)	NO (unless using GPIO add-on module)
GPIO	
5 (1.8Volt)	17 (3.3Volt)
SIZE	
83 x 48mm (3.27" x 1.89")	85.6 x 56mm (3.37" x 2.2")
WEIGHT	
30g (1.06 oz) 48g (1.69 oz) with a heat sink	45g (1.6 oz)
Price	
$59	$35

ning at 1.7Ghz (Can be overclocked to 2.0Ghz). Remember that overclocking is still luck-of-the-draw. Not every board will have the same overclocking result.

Like the RPi, the U3 has a 10/100 Ethernet port. Even so, the U3 shows higher throughput speeds because it has a separated root hub. The ODROID-U3 has 3 USB ports giving more room for the user to connect their desired peripherals. The U3's USB ports can provide up to 500mA each. No reboots or shutdowns are encountered while connecting USB devices.

The U3, unlike the RPi, uses microSD cards instead of a fullsized ones. MicroSD cards are becoming more popular because presently, most smartphones with storage expansion capability on the market today support microSD cards. Also, the U3 goes one step further and adds the option to use eMMC memory. eMMC is a highspeed MLC flash memory like the ones used as the built-in memory on modern phones and tablets (eMMC sold separately as an accessory). The eMMC access speed is 3-4 times faster than SD cards on the ODROID-U3.

The number of GPIO I/O pins is smaller than the RPi. However, you can expand the I/Os with the I2C I/O expander or the USB-I/O kit, in addition to Arduino or Arduino-like interfaces.

Computing Performance comparison

Tests were done on manufacturer-provided images with a clean install, plus apt-get update && apt-get upgrade to ensure that both boards were up-to-date.

The RPi was overclocked at 800Mhz and using a Sandisk Extreme UHS-1 45Mbyte/s 8GB SD Card with the Debian Wheezy OS.

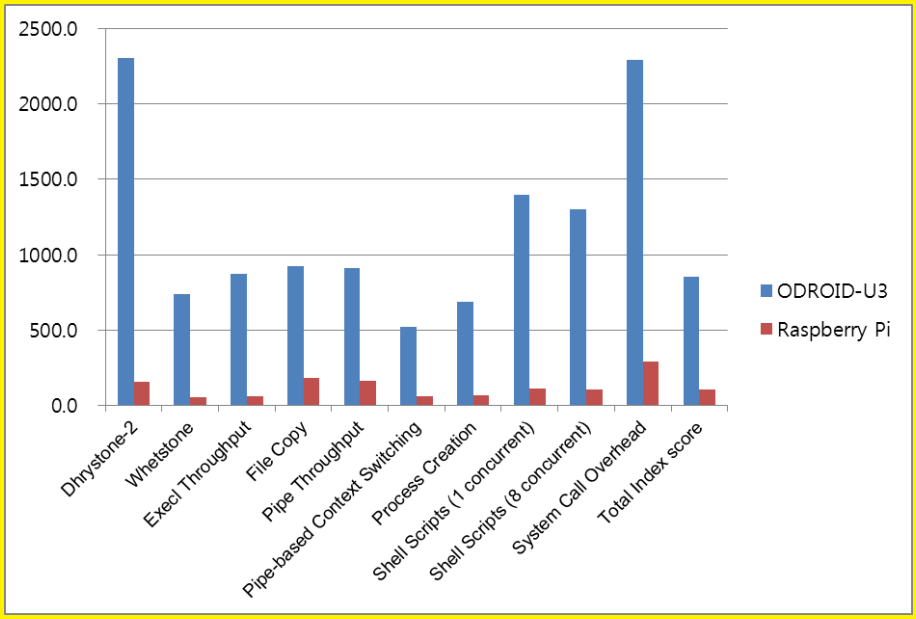
The U3 was clocked at its standard speed of 1.7Ghz and using a 16GB eMMC with Ubuntu 13.10 OS.

Both units were powered by a 5V/2A power supply and connected to the 1920x1080 HDMI output.

We ran a famous and simple benchmark: Unix-Bench version 5.1.3

To use all four cores in the ODROID-U3, we issued the ./Run –c 4 command.

Computing Performance comparison



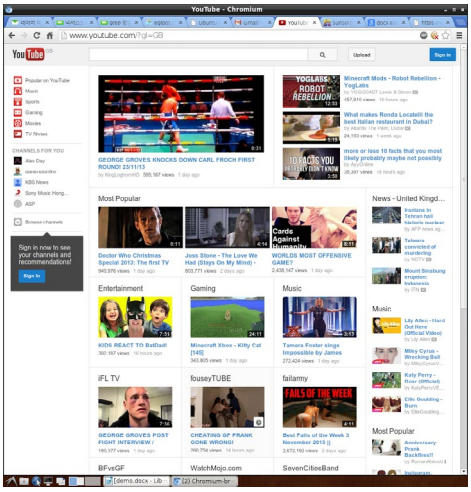
The Dhrystone-2 benchmark is about 14 times faster on the ODROID-U3. File I/O benchmarks at about 5 times faster on the ODROID-U3 because of the faster eMMC storage. For overall performance, the test results show the ODROID-U3 is about 8 times faster than the RPi. However, the price of the U3 is only a factor of 1.7 times that of the RPi.

Daily use Software comparison

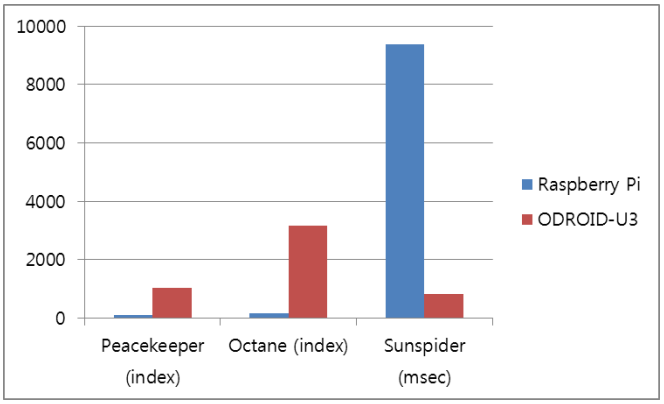
Web Browser:

ODROID U3 provides you a full browsing experience with fluid window scrolling and window moving, whitout ghosting. HTML5 and Flash player are also available on Google Chromium web browser , so that you can enjoy YouTube videos, HTML5 games and more on your U3.

You can enjoy YouTube videos and HTML5 games on the U3.



The U3 gets you pleny of resources to run your browsing needs, the Raspberry, not so much.



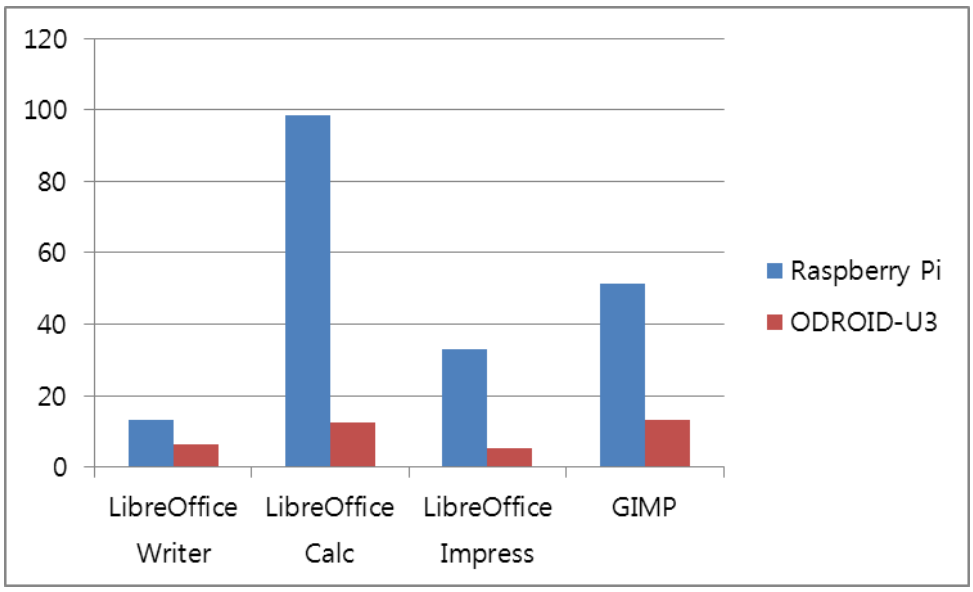
Peacekeeper Benchmark
(Higher is better)
RPi: 99
U3: 1036
Octane Benchmark
(Higher is better)
RPi : 154
U3: 3156
Sunspider Benchmark
(Lower is better)
RPi:: 9372ms
U3: 834ms

U3 shows 10-20 times higher performance compared to the Raspberry Pi.

Office Suite and Photo Graphic editor

Many people use office software frequently. We performed a loading and opening time comparison for files using the famous LibreOffice and GIMP.

We selected a particular file in the file manager to measure the time to open the program plus the file itself.



The most resource-intensive program, Calc, is about 8 times faster on the ODROID-U3.

Software Development

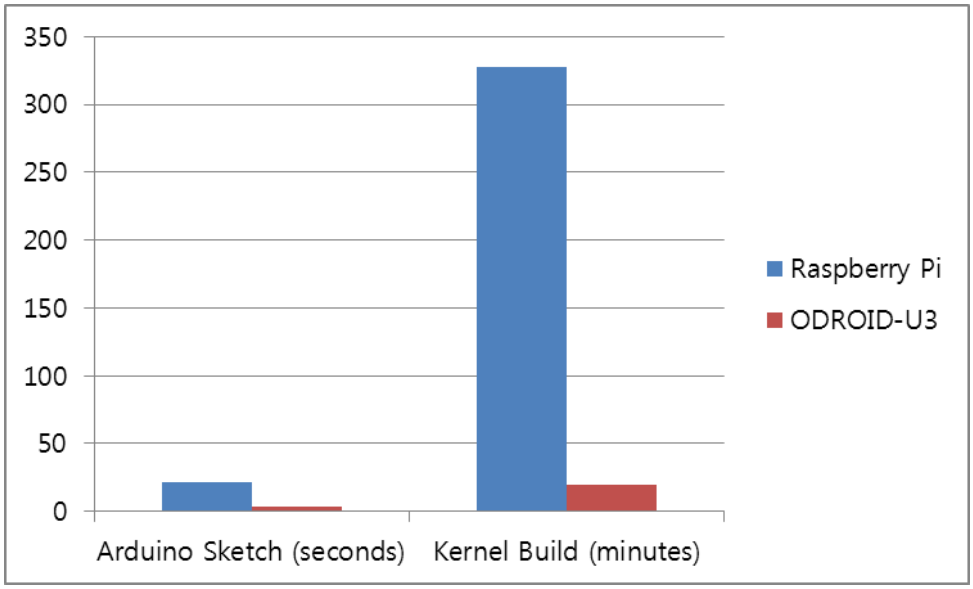
We also compared several IDEs and compilers for software development.

We tested Scratch, Arduino IDE, Python, Linux Kernel building and Eclipse.

The U3 is obviously much faster for loading the IDE and compiling the code while the RPi could not run Eclipse due to insufficient RAM.

The full Linux kernel source building time on U3 is 17 times faster than on the RPi.

We used the –j5 option in the make command to utilize 4 cores.



Arduino Sketch: Build time of a simple LCD demo sketch file.
RPi: 21.7 seconds
U3: 3.2 seconds

Kernel compile: Build time of full Linux kernel 3.4 source code
RPi: 327 minutes (5 hours 27 minutes)
U3: 19 minutes

Entertainment Software and others

XBMC

ODROID offers XBMC support on Linux and Android. Like the RPi, the ODROID-U3 suffers on video decoding when the screen resolution is at 1080p. However at 720p, you'll get a more fluid experience than its competitor, even though work is still in progress on Linux XBMC for the U3. Note that ODROID-U3 doesn't suffer the same video decoding slow-down at 1080p when running Android.

Android OS

Android support is complete with everything working out of the box. The U3 is equivalent to a full HD Tablet when connected to your TV. If you install Google Play on ODROID, you can enjoy over a million Android applications and content. RPi also has an Android port, but it is lacking many features desired for real world usage.

Conclusion

As one can easily see from the above results, the quad-core 1.7GHz ODROID-U3 can easily outperform the single-core 700MHz Raspberry Pi board (even when overclocked). Many test results show six to twelve times more performance from the U3 platform, but at a cost increase of only 1.7 times. Even though both platforms are Linux friendly computing devices, the performance to cost ratio is in an entirely different league.

If you are considering a tiny computer for general purpose computing, software development or as a project platform, the ODROID-U3 will give you a lot more satisfaction and fun with incredible performance for a very low price.

MEET AN ODROIDIAN

A PERSONAL ONE-ON-ONE WITH JUSTIN LEE, CEO OF HARDKERNEL

Rob Roy
@ odroidmagazine@gmail.com

As the CEO of Hardkernel, you still manage to maintain contact with the community through the forums. What do you like most about the ODROID community?

I've been gathering many great ideas, issues, complaints and requirements via our forum. We can improve the hardware and software for ODROID boards with that valuable voice of customers. Because all the topics of our community forum make the future of ODROID, I will manage to maintain contact with the community forum as much as I can.

My favorite forum is "Projects" which shows many brilliant applications. I didn't know well what users can do with ODROIDs. I've been learning many things from the "Projects" forum.

This is a robot project was powered by ODROID-U2 board running ROS with a 3D image depth sensor. Isn't it exciting?

How did you get started with computers?

When I was 13 years old, probably year of 1983, there are many Apple II clone PCBs in Korea. I [had to] solder over 300-400 components on a big Apple II PCB to learn the Basic Language. I spent over 4 weeks to assemble my first PC and finally I got my own build. But I [wasted a] very long time to play games.

Later I learned 6502 ASM language as well as Z80 ASM language with a CP/M card. It was 1984-1985.

What motivated you to start Hardkernel?

Hardkernel was established in late 2008. It is 5 years old. All the founding members had been working for the consumer products [sector with products] like Portable Media player, Ultra-mobile-PC, Car navigation system over 10 years. When the Smartphone devices expanded quickly and killed the other IT devices, we [needed to] find another business.

We thought that many developers need the higher computing performance hardware with the Linux based platform. We made a new company Hardkernel which is simply Hardware + Linux Kernel.

We started to develop a product for developers by developers. The first product was ARM11 based board which runs Android 1.5 in March 2009. The first original ODROID platform was born in October 2009 with ARM Cortex-A8 and

Android 2.0. ODROID means Open + Android.

Who is working with you on the Hardkernel team?

We have 2 teams internally.

Marketing and Sales team is performing Marketing, Financial, Sales, Web management and other operation activities. Lisa, Anna and Emily are one team.

Research and Development team is performing Architecture design, Hardware schematics and PCB drawings, Software development, Production, Shipping and Customer supports. Charles, Ruppi, Kevin, Mauro, Chris, David, John and Brandon are one team.

The Hardkernel team, to know more about the story of this photo, take a look at our blog!

Do you have any personal projects that you're working on using ODROIDs?

I'm using an ODROID as a game console for my family. One day I want to make a real Arcade cabinet with a coin detector to play exciting games with my colleagues in our office.

What is coming up next for Hardkernel and ODROID in 2014?

As we did for last 5 years, we will keep developing the platform for software and hardware.

We've just launched the ODROID-U3 for various projects with a highly affordable price. There will be many exciting peripherals and software for ODROID-U3 in 2014. All the ideas are welcome and we are ready to make it. I hope we can show you a couple of peripherals every month.

We will write a couple featured articles for The ODROID Magazine every 2 months. It is the most important plan in 2014.

What other interest and hobbies do you enjoy?

From spring to autumn season, I'm playing in my small farm every weekend. I plant corn and a few different vegetables such as tomato, chili peppers and cabbages. It seems to be really good for my health as well as my brain refreshing and reducing mental pressure.

Thank you for taking the time to speak with us, Justin, and we wish your continued success with Hardkernel!

Hardkernel's first prototype, a long way has passed from this, join us to know what's next!