Arx Fatalis • CEC on ODROID-C1 • Open GPS • Retro Gaming Console

ODROID Magazine

Year Two
Issue #17
May 2015

N64 Emulation

How to run classic games on all of the ODROID boards

- ARM Solar Challenge
- Tiny Linux Box
- Quake II on ODROID-C1
- Observe a Solar Eclipse with Sensors

Wall Dash Board

What we stand for.

We strive to symbolize the edge of technology,
future, youth, humanity, and engineering.

Our philosophy is based on Developers.
And our efforts to keep close relationships with
developers around the world.

For that, you can always count on having the quality
and sophistication that is the hallmark of our products.

Simple, modern and distinctive.
So you can have the best to accomplish
everything you can dream of.

HARDKERNEL

Pollin ELECTRONIC

We are now shipping the ODROID-U3 device to EU countries! Come and visit our online store to shop!

Address: Max-Pollin-Straße 1
85104 Pförring Germany

Telephone & Fax
phone : +49 (0) 8403 / 920-920
email : service@pollin.de

Our ODROID products can be found at
http://bit.ly/1tXPXwe

EDITORIAL

Gaming is one of our favorite ODROID pastimes, with dozens of emulators available on both Linux and Android. Tobias, our celebrated Linux gaming columnist, evaluates some of the best Nintendo 64 games available on a C1, U3 and XU3 to find out which platform is most suitable for smooth game play. We also take a look at Arx Fatalis, Quake II, Does Not Commute, and Hearthstone, as well as a slick method of re-using a PlayStation 2 as a unique case for an ODROID-C1 emulation station. Venkat, our resident technical guru, details creating your own GPS tracking system, capable of monitoring fleets of vehicles for your business, and Pascal shows us how his ODROID was able to see a solar eclipse on a cloudy day using its electronic sensors. If you enjoy DIY projects, check out our guides to creating a control room dashboard for keeping track of metrics and data in real time, and building a tiny Linux box disguised as a standard wall charger that you can take anywhere.

ODROID Magazine, published monthly at http://magazine.odroid.com, is your source for all things ODROIDian.
Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815
Hardkernel manufactures the ODROID family of quad-core development boards and the world's first ARM big.LITTLE single board computer.
For information on submitting articles, contact odroidmagazine@gmail.com, or visit http://bit.ly/IyplmXs.
You can join the growing ODROID community with members from over 135 countries at http://forum.odroid.com.
Explore the new technologies offered by Hardkernel at http://www.hardkernel.com.

HARDKERNEL

ODROID Magazine

Rob Roy, Chief Editor

I'm a computer programmer living and working in San Francisco, CA, designing and building web applications for local clients on my network cluster of ODROIDs. 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 use my ODROIDs for a variety of purposes, including media center, web server, application development, workstation, and gaming console. You can check out my 100GB collection of ODROID software, prebuilt kernels and OS images at `http://bit.ly/1fsaXQs`.

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-U3. Regarding hobbies, if you need some, I'd be happy to give you some of mine as I have too many. That would help me to have more time to spend with my wonderful wife of 23 years and my four beautiful children.

Bruno Doiche, Senior Art Editor

Last time we've seen this mischief of an Art editor, he was learning how to do the secret sauce to perfect his Entrecôte, doing about 2 hours of exercise daily and scouting all bike shops at his home town looking for that guy that shares his passion for cycling to be the mechanic for all his bikes.

Nicole Scott, Art Editor

I'm a Digital Strategist and Transmedia Producer specializing in online optimization and inbound marketing strategies, social media directing, and media production for print, web, video, and film. Managing multiple accounts with agencies and filmmakers, from Analytics and Adwords to video editing and DVD authoring. I own an ODROID-U3 which I use to run a sandbox web server, live in the California Bay Area, and enjoy hiking, camping and playing music. Visit my web page at http://www.nicolecscott.com.

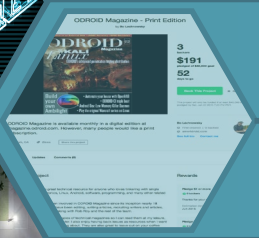
James LeFevour, Art Editor

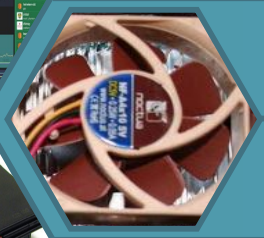
I am on a sabbatical and will return to working on ODROID magazine this summer.

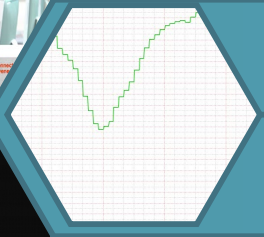
Manuel Adamuz, Spanish Editor

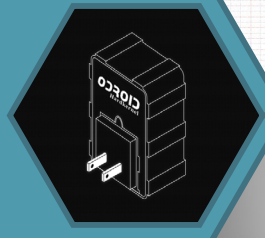
I am 31 years old and live in Seville, Spain, and was born in Granada. I am married to a wonderful woman and have a child. A few years ago I worked as a computer technician and programmer, but my current job is related to quality management and information technology: ISO 9001, ISO 27001, and ISO 20000. I am passionate about computer science, especially microcomputers such as the ODROID and Raspberry Pi. I love experimenting with these computers. My wife says I'm crazy because I just think of ODROIDs! My other great hobby is mountain biking, and I occasionally participate in semi-professional competitions.

INDEX









ARX FATALIS

A LONG-AWAITED GAME MAKES A STAR APPEARANCE

by Jeremy Kenney

Arx Fatalis is now available on the ODROID as Arx Libertatis

I have a lot of games on my shelf, and one of them that stands out as a favorite is a Diablo-style role playing game (RPG) with first-person shooter (FPS) elements. Arx Fatalis, which is now available as an open-source version under the name Arx Libertatis, rewards you with lots of action and looting. You can customize your character using one of four different body types, and there are even some Minecraft-inspired gameplay features that involve cooking, making weapons and customizing them. You can also cast spells by drawing shapes according to runes picked up along the way by learning them from the skillbook. The action itself is a bit slow-paced at first, but you get stronger and faster as you progress. You should save your game often since it does not have an autosave feature, and because it has lots of action, you can die quickly.

The graphics still hold up well today, but there are some occasional glitches, such as such killing a rat and seeing under his jaw, although it's easy to look past them because of the great gameplay. When running 720p or better, the game looks even better than originally intended, which is a big plus. The textures feel as if they were upscaled by a professional Photoshop expert. With some easy steps, you can be running Arx Liberatis in just a few minutes!

Arx Fatalis has collectable runes which are used to cast spells

We will need to have the LibGL and LibGLEW libraries, as provided by @meveric in his repository. If you haven't added his repository yet, follow the steps below to access it. If you have an ODROID-U2 or U3, launch a Terminal window and type the following, making sure to provide the root password for your system when requested, which is usually "odroid":

```
$ su
# cd /etc/apt/sources.list.d
# wget http://oph.mdrjr.net/meveric/sources.
lists/meveric-all-U.list
# wget http://oph.mdrjr.net/meveric/sources.
lists/meveric-all-testing.list
# wget http://oph.mdrjr.net/meveric/sources.
lists/meveric-wheezy-main.list
# wget http://oph.mdrjr.net/meveric/sources.
lists/meveric-wheezy-backports.list
# wget http://oph.mdrjr.net/meveric/sources.
```

Spells are cast by moving the mouse around to match the rune

```
lists/meveric-wheezy-testing.list
# wget http://oph.mdrjr.net/meveric/sources.lists/
meveric-all-XU3.list
```

Download and install the repository's signature key in order to notify the "apt" application that packages signed with the key are safe to use, then update the local repository information:

```
# wget -O- http://oph.mdrjr.net/meveric/meveric.asc |
apt-key add -
# apt-get update
```

Once the update completes, you can install the required libraries:

```
# apt-get install libgl-odroid
```

Next, download my Debian pre-prepared package from MEGA by visiting `http://bit.ly/1OSKMDq` using any web browser and pressing the "Download" button. After the file has been completely saved, return to the Terminal window to install the game:

```
# cd ~/Downloads
# dpkg -i arx-libertatis_1.1.2-1_armhf.deb
```

There is a launcher icon that will appear in your App Launcher bar after the installation completes, but the configuration directory and files need to be generated before installing the data files. Type the following into a new Terminal window as a normal (non-root) user, which generates the configuration (.cfg) files:

```
$ arx
```

The final step is to install the data files. If you happen to own a copy of Arx Fatalis, insert the CD into any computer with a CD or DVD drive and copy the following files to a USB drive, then insert the USB drive into the ODROID, and copy the data files into `~/.local/share/arx`:

```
data.pak
data2.pak
speech.pak
sfx.pak
loc.pak
```

If you don't own the CD, you obtain the shareware version of the game on the Arx Liberatis website at `http://bit.ly/1GvseKm` under the Demo section. Unzip the downloaded file, locate the following files, and copy them into `~/.local/share/arx`:

```
data2.pak
loc.pak
```

Once the data files are installed, navigate to the `~/.config/arx` directory and edit the cfg.ini file in order to update the configuration. If you're using an ODROID-U2 or U3, you can edit the fullscreen features to run at either 1280x720 or 1920x1080, which can also be configured inside the game. If you're using an ODROID-XU3, set the resolution to windowed, then save the file. Now you're ready to run the game and play it!

Using the Terminal window, it's necessary to include a reference to LIBGL when running Arx Fatalis. The following command will launch the game with the necessary libraries enabled:

```
$ LD_LIBRARY_PATH=/usr/local/lib LIBGL_FB=0 arx
```

At this point, Arx Fatalis should be running. On the ODROID-XU3, there is some minimal tearing when performing 360 movements extremely fast, but other than that, the graphics are perfect. The game is very stable, fun to play, and lasts quite a while, although I wish that it supported multiplayer modes with cooperative quests. The voice acting is also very high quality for people who like to follow the storyline. I am proud to include this type of game in my ODROID collection, since a lot of us enjoy an FPS/RPG hybrid game with blasting action and awesome special effects.

If you'd like to learn more about the open-source version of Arx Fatalis, please visit the website for Arx Libertatis at http://bit.ly/1PHMoAy, or download the source code from GitHub at http://bit.ly/1L3aHIu.

ODROID MAGAZINE KICKSTARTER CAMPAIGN

GET THE DELUXE PRINTED VERSION SENT TO YOUR HOME

by Bo Lechnowsky

My name is Bo, and I've been involved in ODROID Magazine since its inception nearly 18 months ago. My main tasks have been editing, writing articles, recruiting writers and articles, and promoting the magazine along with Rob Roy and the rest of the team. I personally like to have paper copies of technical magazines so I can read them at my leisure, even when I am not near a computer. I also enjoy having back issues as resources when I want to do something I remember reading about. They are also great to leave out on your coffee table, your office's waiting area or break room, or your local makerspace.

Unfortunately, printing and shipping a paper-based color magazine worldwide is not free, or even cheap. The idea of bringing ODROID Magazine to the public in a print edition is a strictly not-for-profit effort, and the digital version will still be free. According to the quotes we have at present from the publishers and the United States Postal Service, $40,000 is the minimum it will cost to print and ship 1,000 copies of the magazine for 12 months. We're not going to make any profit on the print edition of the magazine, and may actually lose some money. If we end up with extra money from advertising, then we we will be lowering the subscription price to make it more affordable.

If you're like me, and you'd like a print magazine subscription, consider supporting the project with a $35.88 subscription for one year. This equates to $2.99 per issue, which includes postage costs within the US. Or, if you have a product that is related to the topics covered in the magazine, consider supporting the project by signing up for one of the advertising benefits, or do both. You don't need to back the Kickstarter if you want to subscribe at a future date, but by backing this project on Kickstarter, it will help us to launch this project so you have the option to subscribe at a future date. Help us spread the ODROID love to the world!

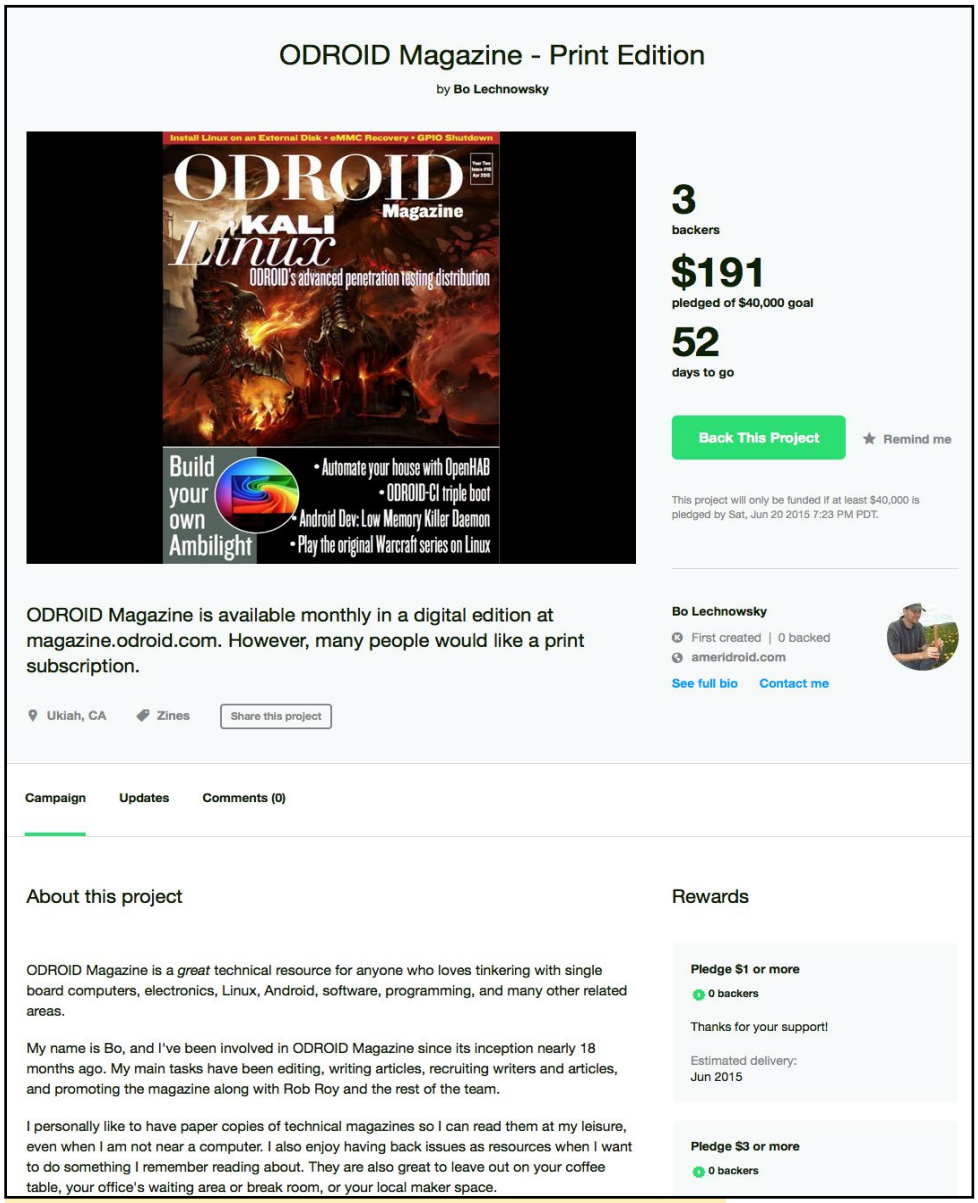
Risks and challenges

My efforts involving the magazine have been restricted over the past six months or so as I've been involved in expanding ODROID distribution in North America through ameriDroid.com.

However, this has also given us the unique advantage of having a facility and staff that is already devoted to handling the distribution of many thousands of ODROID orders monthly to locations around the world, but with a high concentration in the US and Canada. Our best calculations suggest that dealing with print magazine and subscription-related issues (handling, mailing, returned issues, cancellations, emails, phone calls, etc.) will take about $1,100 in total personnel costs per month.

The largest risk is in contracting a print house to produce the magazine at a reasonable price. We've already been in contact with two such companies in our area and have received quotes from them. Per-issue pricing only starts to make sense at 1,000 copies per print run due to set up costs, so we need to have at least that many subscribers to make this work. Our most reasonable estimate is that 1,000 copies of a 48-page magazine will cost $1,496 per month plus $123 in tax on 35# 80 Brite paper with a 35# 80 Brite cover.

Another risk is in successfully completing the formal application procedure and the nonrefundable application fee of approximately $500 to become authorized for Periodicals mailing privileges through the United States Postal Service. However, the publishing companies have a lot of experience in going through this application procedure, and we have a great personal working relationship with our USPS District Manager and our local Postmaster. Assuming all 1,000 copies are sent to only the United States (the cheapest postage), the costs associated with this are $150 per month for postal paperwork fees (not including the periodicals permit) and labeling and $794 in postage.



Click here to help create a printed version of ODROID Magazine

Adding the above costs together:

$1,100+1496+123+150+794=$3,663 per month. **$3,663/** month for a year comes to **$43,956**. Kickstarter and payment processing fees total 8-10% of the successfully funded campaign, so that will reduce the actual amount for this project to somewhere between **$36,000** and **$36,800** if it is minimally funded. ameriDroid.com is willing to underwrite the difference, up to **$7,956**, if we don't get enough additional support to cover that.

If we get advertisers for the print edition, that may potentially help lower the cost of future subscriptions and per-issue prices. All advertisements are planned to be included after the content that was included in the digital edition, and not interspersed throughout the magazine.

To sign up for a printed subscription to ODROID Magazine, or to donate money to the campaign, please visit `http://kck.st/1EScVeu`.

KICKSTARTER

DOES NOT COMMUTE

THE CRAZIEST DRIVING GAME AROUND

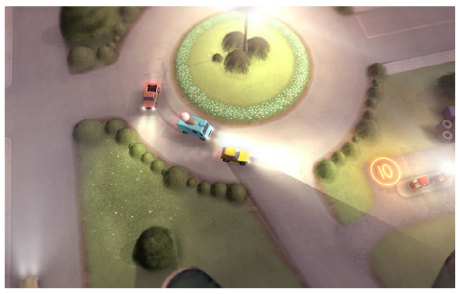
by Bruno Doiche

Have you ever wondered what would happen if you could combine missions from the original Grand Theft Auto with the replaying madness of Super Meat Boy? Install Does Not Commute on your Android and start with the first task of going from point A to point B, then go through your previous moves over and over until you are in the greatest mess of a traffic that you could possibly imagine.

Not only will you have fun, you should really expect to go insane with this game!

`https://play.google.com/store/apps/details?id=com.mediocre.commute`

As described by the publishers, Does Not Commute is a temporal paradox in which you have no one to blame but yourself.



TRANSFORMERS BATTLE TACTICS

A FUN WAY TO PLAY AGAINST YOUR FRIENDS IN EPIC ROBOT BATTLES

by Rob Roy

Transformers started out as real car and truck models that could be changed into robots with a few twists and turns. They eventually were turned into a Saturday morning cartoon, and most recently, a series of larger-than-life Hollywood blockbusters. Now you can play as a Transformer on your ODROID device using a mouse and keyboard in a unique PVP turn-based battle game. Part of the strategy is knowing exactly when to transform into a vehicle for special battle advantages!

https://play.google.com/store/apps/details?id=com.dena.west.TransformersBattleTactics

Transformers Battle Tactics brings to life many of your favorite Transformer characters in an epic PVP battle arena

FIXING CONSUMER ELECTRONICS CONTROL (CEC) ON THE ODROID-C1

ENABLE REMOTE CONTROL VIA HDMI

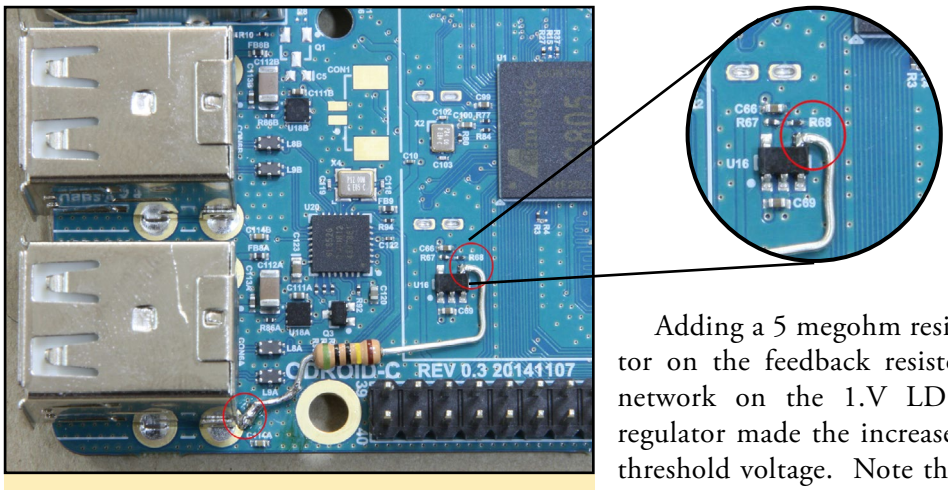
by Justin Lee

The ODROID-C1 includes a Consumer Electronics Control (CEC) option in its base hardware, which allows control of certain applications using a TV remote. CEC is used to send signals to the board via the HDMI cable, which can then be used to trigger keyboard and other events. However, there have been many reports of the CEC functionality not working reliably, and Hardkernel has tried to address the issue, even though the board was never advertised with CEC availability. It was eventually discovered to be that the threshold voltage level of CEC input pin on the S805 CPU is too low and can't read the CEC signal in a stable way. The threshold level is not adjustable by software, and Hardkernel has not been able to fix the issue with a driver update. The main problem is that logic-level 0 can't be read correctly due to the offset of the CEC signal, which is explained in detail at `http://bit.ly/1ItPWWM`.

To make the CEC signal more stable, it is necessary to remove the NXP IC and add a few discrete components in order to implement the level shifters. However, it seems to be almost impossible due the limited space and some tiny components. Hardkernel engineers then spent a few more days of investigation in order to find a better way, and came up with the following method of fixing the CEC hardware. If you or your friend can do soldering well, you can try this modification. If you have a yellow sticker on the Ethernet port, this fix

Adjusting the input-low threshold

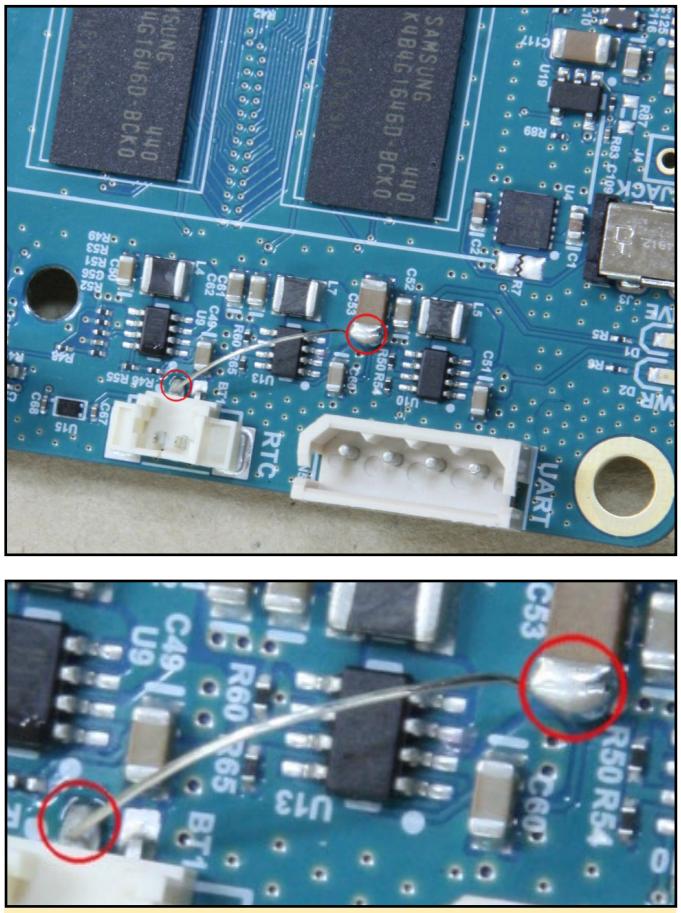
According to experimental tests, if the VDDIO 1.8V signal is increased to approximately 1.9V, the input-Low-threshold voltage can be increased as well. We modified several boards and we observed all of them could read the CEC signal correctly. Here we show an ODROID-C1 board after modification.



Adjusting the VDDIO with a resistor

Adding a 5 megohm resistor on the feedback resistor network on the 1.V LDO regulator made the increased threshold voltage. Note that the 1.8V rail is also used for some other blocks on the board, so we could only increase the signal less than 7~8% in order to minimize any negative side effects. A 1% tolerance resistor of 4.99 or 5 megohms must be used.

Another important fact is that the CEC block in the S805 CPU must be powered via the RTC power rail, which means that you must install the RTC backup battery to activate the CEC block. If you don't have an RTC backup battery, you will need to solder a wire line, as shown here. The RTC battery current consumption increased only about 25% when we pressed the remote controller buttons continuously. Please note that if you have the RTC battery, the wire should not be added.



Connecting the RTC to activate the CEC block

To test the CEC functionality after modification of hardware with the RTC power, we used the Android operating system. When any of the 4 arrow keys were pressed on the remote controller, the GUI on the Android reacted to the CEC input. However, it works with only Samsung models, while a similar LG model didn't work.

It is also necessary to modify the boot.ini file, which is located on the FAT32 boot partition, by adding the line in bold shown below, in order to activate the CEC functionality:

```
setenv hdmimode 1080p
setenv cecconfig cecf
setenv bootargs "root=/dev/mmcblk0p2 rw console=ttyS0,115200n8
no_console_suspend vdaccfg=${vdac_config} logo=osd1,loaded,${fb_
addr},${outputmode},full hdmimode=${hdmimode} cvbsmode=${cvbsmode}
hdmitx=${cecconfig} androidboot.serialno=${fbt_id#}"
setenv bootcmd "movi read boot 0 0x12000000; movi read dtb 0 0x12800000;
bootm 0x12000000 - 0x12800000"
run bootcmd
```

Normally, the warranty would be violated once you solder the board. However, we will make an exception for anyone performing this CEC modification. We have also implemented a customer program for any ODROID-C1 owners who require CEC functionality but cannot solder the board themselves. Please contact Hardkernel at odroid@hardkernel.com if you would like to participate in the program. **If there is a yellow sticker on the Ethernet port of your ODROID-C1, the CEC modification has already been done at the Hardkernel factory.**

If you have questions, comments, or suggestions, refer to the original CEC announcement on the ODROID forums at `http://bit.ly/1ziyZMU`.

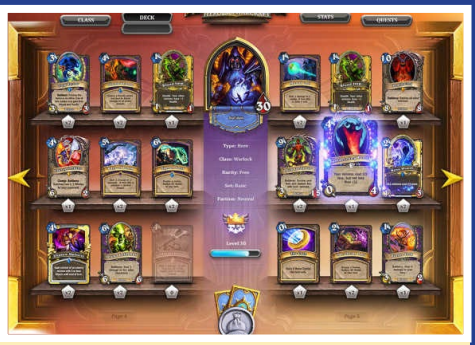
HEARTHSTONE HEROES OF WARCRAFT

THE GREATEST ELECTRONIC CARD GAME TO DATE

by Bruno Doiche

Since I can remember, I always wanted to play Magic the Gathering on my computer, but the online version never managed to capture the excitement of a good one-on-one card game. Then, I started playing Hearthstone, which is published by Blizzard Entertainment. If you like card games, and enjoyed playing the massively popular World of Warcraft series, you can't go wrong with this game. Just promise yourself not to spend all of your allowance money on Hearthstone cards!

`https://play.google.com/store/apps/details?id=com.blizzard.wtcg.hearthstone`



Hearthstone combines the excitement of Magic the Gathering with many favorite World of Warcraft characters

LINUX GAMING: NINTENDO 64 EMULATION - PART I

EMBARK ON THE ULTIMATE 90S GAMING JOURNEY

by Tobias Schaaf

Nintendo 64 emulation has recently evolved to run on all ODROID devices, using either the Mupen64plus standalone emulator or the Libretro core for Retroarch. Now that it's widely available, I decided to do a comparison not only between the standalone version and the Libretro core, but also between the different ODROID platforms, in order to evaluate their capabilities in terms of emulating a Nintendo 64 (N64) console. Please note that this article covers only Linux emulation, and does not extend to Android, although there are several Nintendo 64 emulators available for Android, such as Mupen64plus and N64oid.

General information

It took a while to get N64 emulation to work on all the ODROID boards under Linux. However, now that it's functioning, it's quite fun and opens up lots of opportunities for classic gaming. Hopefully in the future, we will see more improvement and have even better support for N64 emulators on ODROID devices under Linux. For now, there are a few restrictions. Only the XU3 is able to use the Libretro core under Linux, which has better graphics, and is easier to control than the standalone Mupen64plus emulator. Mupen64plus runs on all other ODROID devices, such as the Exynos 4 series (X, X2, U2, and U3) as well as the newer but less powerful ODROID-C1. Both versions offer different plugins and methods of playing the games.

Graphics plugins

Whether you use Mupen64plus or the Libretro core, different plugins are used to display the game graphics. Mupen64plus is able to use a video plugin called rice, and another one called glide64mk2. The Libretro core offers rice, glide64 and one called gln64. While testing, I found that the best videocore depends on the game. However, it seems that glide offers better graphics features than the rice plugin, but has some minor glitches that are not present in the rice video plugin.

Using the standalone Mupen64plus, rice is unable to perform aspect ratio scaling, and always scales the game to the full size of your video resolution. This distorts the picture, causing the characters and objects to appear wider than normal. The ODROID-C1 performs best when using the rice video plugin, since glide64mk2 doesn't work unless the color depth is reduced to 16 bit, which causes the transparency effects to become disabled. This will also cause issues if you try watching movies or want to start other applications that require more than 16 bit color depth. Since the initial tests on the C1 went poorly, I decided to retest every game in 16 bit using the glide64mk2 video core. There seems to be a workaround using framebuffer drivers instead of X11 drivers by adding some scripts in order to switch resolution and color depth, but since

my ODROID GameStation Turbo image uses X11 drivers by default, I don't take the time to perform framebuffer tests.

The glide plugin on the Exynos 4 series devices (X, X2, U2 and U3) is working well, and respects aspect ratio with an overall good quality, but it can be a bit slower than rice on some games. Glide also seems to render a darker picture than rice does, which is most likely due to some missing shader options with regards to shadows. The glide64mk2 plugin on the Exynos 4 devices is the preferred graphics plugin for Mupen64plus standalone. The XU3 can use rice, glide64 and gln64, but glide64 seems to be the best plugin for now on the ODROID.

Controllers

Joysticks are fortunately working fine on all ODROID devices, which means that all emulators (mupen64plus and libretro core) are fully supported with any game controller. The Mupen64plus emulator configures the controllers automatically, but not all controllers work perfectly with the default settings. Thanks to Retroarch on the XU3, you can setup any controller you want by manually configuring the buttons, so every controller is 100% supported.

Normally you should be able to activate rumbling support of the controller, but I had trouble getting it enabled on all emulators and controllers. I was able to use it with some PS3-style controllers on the Mupen64plus standalone emulator, but I wasn't able to use rumbling with the Libretro core.

Sound

Sound is working well on all emulators, and I haven't found any major issues with it. Although one game had a delay in sound, which caused effects not to be synced with the action on the screen, that was an exception, and I haven't seen this issue with any other game.

Game selection

Are you ready to play your favorite Nintendo 64 games on the ODROID? Well, that's exactly what we want to try and find out: do your favorite games work? To answer that question, I did some research on what are generally considered the best games on the N64, then picked some of them to test, as shown in Figures 1 and 2. Hopefully you will find some of your favorite games in this list as well.

A list of some favorite N64 games from many different genres

Banjo-Kazooie

Banjo-Kazooie is a mix of jump-and-run platformer and action adventure. You play as Banjo the bear trying to save his sister who was abducted by a witch. He has a friend Kazooie the Bird, with whom you need to solve a couple of puzzles. Like most of the Rareware games, this one is fun and has a cute comic style.

U3

Generally, the game is running acceptably on the U3. It's sometimes a little laggy, especially during the intro. The introductions of Rare games are normally rather long and can't be skipped. On the XU3 and the Libretro core, you have the ability to increase the emulator speed, so that the introduction is over faster. I haven't seen that option on the standalone Mupen64plus emulator yet, which means a long wait time. Also, the fonts are not correct on the standalone emulator, which is slightly annoying.

The game felt a little laggy after playing it for a while. I used the configuration options of the emulator to activate frameskipping with a maximum of three frames, which increased gameplay to full speed. With that setting, it was a really nice game to play, with only the font issue remaining. I left the frameskipping option set to three frames for all of the other games.

C1 - rice plugin

For the C1, I used the standalone emulator Mupen64plus, using the rice plugin, since I did not want to change the colors to 16 bit. Also, rice is a little bit faster than glide64mk2, and is better suited for the C1. I also had to activate frameskip for the rice plugin in order to get the game to run smoothly. Without frameskipping, the sound was lagging and was not a good experience.

Although rice does not respect aspect ratio, it doesn't look bad. The issues with the fonts that happened with glide64mk2 do not exist on rice, so the fonts look normal. With frameskip activated, the game ran surprisingly fast on the C1, which was unexpected. If the game were to support a proper aspect ratio, it would run perfectly fine on the C1.

C1 - glide64mk2

Banjo-Kazooie runs on the ODROID-C1 using glide64mk2, but is extremely slow and no fun to play. The Rice plugin in 32 bit color is working much better than glide64mk2 in 16 bit.

XU3

XU3 uses the Mupen64plus Libretro core and Retroarch to emulate the game, and the experience on the XU3 is the best of all three platforms. The emulator runs glide2gl as a video plugin, which seems to be much better than the older glide64mk2, and does not render the colors as darkly. The Libretro core is missing the frameskip feature that the Mupen64plus emulator offers, which means that it can only perform as fast as the board that it runs on, which can lead to slowdowns, depending on the scenes. In Banjo-Kazooie, this happens in the introduction, but it's not bad. The graphics look much better using Libretro, and the game is fully playable.

Conker's Bad Fur Day

Conker's Bad Fur Day is another game from Rareware, and is similar to Banjo-Kazooie. However, it's not suitable for small children due to its reference to drugs and alcohol along with harsh language, despite its comic style. You will also find a few characters that are the same in both games.

This game is a mixture of many genres, mostly jump-and-run and action adventure, but it feels more like a first-person shooter with a mix of other genres as well. The game is actually one of my favorites for the Nintendo 64, and some other people already gave it nice reviews: http://bit.ly/1bo6odW. I highly recommend the game for adults and teens.

U3

The U3 with the Mupen64plus standalone emulator is a little bit too slow for Conker's Bad Fur Day, and there are scenes where it feels somewhat laggy, which affects the controls. Sometimes they react too slowly, which is annoying during the jumping puzzles. The glide64mk2 plugin also makes the graphics very dark, especially during the cutscenes in the castle. When inside dark rooms, it's nearly all black in some spots.

U3 - rice plugin

While the U3 is having speed issues, the C1 fails completely because the C1 is simply not powerful enough to run a demanding game like Conker's Bad Fur Day. The rice graphics plugin also has many issues with this game, such as black borders and distorted graphics, which is not fun to look at. Although the game is generally working, it's rather slow. Some scenes are actually fast enough to be considered playable, but are far from full speed. Therefore, I would consider this game unplayable on the ODROID-C1.

U3 - glide64mk2

Conker runs better using the glide64mk2 plugin than with the rice plugin. It's still not full speed, but if you can tolerate a little lag, it's playable.

XU3

The XU3 offers the best gaming experience when running Conker's Bad Fur Day. The game, although not running at full speed, is mostly smooth on the XU3. The glide2gl plugin looks really good and only has a few issues with the game. I am not very far in the game right now, so I can't compare how the later levels perform, especially with driving tanks and using sniper modes.

Earthworm Jim 3D

Earthworm Jim is a nice platform action shooter about an earthworm named Jim, who got hit by an advanced space suit, transforming him into a hero. While the game was first a success on the SNES, SEGA Genesis, and even the Playstation 1, with the N64, it went a step further and transformed the game from a 2D platformer into a 3D action game.

U3

The gaming experience of Earthworm Jim 3D on all ODROIDs is very nice. The U3 runs the emulator very fast and fluently, with some occasional minor graphical glitches. Since the game is rather colorful and bright, with light rooms and no shadows or dark corners, the glide64mk2 graphical darkness that affects other games is not a factor while playing Earthworm Jim 3D, which really improves the gaming experience.

C1 - rice plugin

Although the introduction and demo gameplay are fast, I couldn't start the game. The first scene where you talk to one of your friends is not only laggy, but also the window that is supposed to show the text remains empty and does not render anything. Clicking a button is also unresponsive, so you're stuck before the game even starts. It's likely that this is only a rice bug, but since I haven't tested glide64mk2 on the C1, I can't tell how well it's performing under that plugin. Therefore, I can only say that Earthworm Jim is not working on the C1.

C1 - glide64mk2

This game runs really well using the glide64mk2 plugin, with no graphical issues or slowdowns. Everything works as expected.

XU3

Since the C1 and U3 are powerful enough to play the game fluently, it's really no surprise that the gaming experience on the XU3 is perfect as well. If you like a good action platformer, this game is definitely a must have, but it does have some minor glitches. Some of the objects that you can collect are not displayed correctly. They seem to be too high, and you often only see a shadow where the object is supposed to be. You can still collect them, they are just invisible, although the game still works fine otherwise.

GoldenEye 007

GoldenEye 007 is found on everyone's top 10 list for the N64, since the game was revolutionary for it's time. Not only did it offer nice graphics, but it was known for its awesome multiplayer mode. The single player story and missions are very exciting as well and are fun to play as well. GoldenEye is a rather serious first person shooter with just the right touch of secret agent work. Although not as spectacular as Cate Archer, James Bond fights or sneaks through different levels and has to defend himself against enemy guards and spies. However, Cate Archer will always be my favorite spy in harms way. Although this game has very good reviews for the N64, I really don't like first-person shooters on consoles. Therefore it's not one of my favorite games, although it's nice to play.

U3

The game runs very well on the U3. Except for a short scene in the introduction, there was no slowdown either inside or outside of buildings. I had some issues when using a wireless XBox 360 controller with the right analog stick, which made game movement difficult. However, using only the left analog stick seemed to be a good workaround for playing the game.

C1 - rice plugin

The C1 has graphical issues with the rice plugin with this game as well. Neither the logo, nor the introduction are visible, and both are hidden behind a black border. The scene that caused a massive slowdown to 8 fps on the U3 is too much for the C1, and the emulator stops completely and eventually crashes. Observing the ODROID while running the game, I can tell that when the slowdown happens, the RAM usage skyrockets to the point where no RAM is available. After that, it uses the swap partition that I created, and after that, the out of memory killer terminates the emulator, which doesn't happen on the U3. I then switched to the rice plugin on the U3, and although the emulator was much slower than with glide64mk2, it was working properly with no black screen or memory issue, and did not crash. I therefore concluded that it was only an issue with the C1.

C1 - glide64mk2

The game is working with glide64mk2, but the speed varies from nearly full speed to laggy. It's playable, but not as good as on the U3 or XU3.

XU3

The game runs at a decent speed on the XU3 using the Libretro core. The graphics look really good, but it has occasional slow downs, although not in a way that prevents playing the game. I was also able to use the XBox 360 controller without any issues.

Kirby 64: The Crystal Shards

Who does not recognize little Kirby? This game is very kid friendly and has cute graphics. The pink marshmallow-like buddy can suck in its enemies and swallow them in order to absorb their powers. The N64 version has beautiful 3D graphics and is rather easy to play, which makes it perfect for children. Although the game is rendered in 3D, the levels are very linear. You can go left, right, up and down, but are not able to walk freely on the map, which probably greatly reduced the map size and allowed extra performance for effects. The game looks similar to Mario 64, but without the free movement in all directions.

U3

The general experience is very good, and the game runs perfectly fine in full speed. However, it has some graphical glitches with the ground and shadows which makes it flickery in some situations. I also had major issues with the controls. I had to change to a different controller, since my XBox 360 controller would not work with this game. It seems that movement only works with the D-Pad, which is not available on the Xbox 360 joystick.

After I switched to a "Thrustmaster Dual Trigger 3 in 1" controller, which is similar to a PS3 controller, the movement controls were working, and I could play the game without issues besides the previously mentioned glitches. The game experience is really smooth and fun to play. The cute graphics and cutscenes really fit the game.

C1 - rice plugin

The C1 experience was different from the U3, and was unexpectedly slow. Cutscenes were so slow that I skipped them, rather than waiting for them to play through. However, the graphical glitches were gone. The gameplay was slower than expected and was laggy in some scenes, while during other scenes, it's nearly full speed. I expected it to run better on the C1.

C1 - glide64mk2

Kirby works well using glide64mk2, with only minor issues on the ground textures and shadows.

XU3

The XU3 runs Kirby very smoothly. The graphics, although not as bad as on the U3, have some glitches such as effects that do not display on the ground, but they are shown normally while jumping or on higher platforms. Since the XU3 can use the Libretro core, there was no issue with the controller at all, and I was able to use an XBox 360 controller normally.

Next issue - more Nintendo 64 game reviews, including Mario Kart, Mario Party and Paper Mario!

OPENGTS

A POWERFUL OPEN-SOURCE GPS TRACKING SYSTEM

by Venkat Bommakanti

OpenGTS is a highly capable open-source web-based GPS tracking services framework for vehicles, produced by GeoTelematic Solutions, Inc. Although it is useful for a single car or truck, it really excels when used to manage fleets of vehicles. It organizes them into groups of user-chosen types, and the location of each vehicle can be mapped instaneously, with dynamic plotting of its travel route. Speeds may also be monitored in real-time, and unnecessarily long breaks can be flagged. Performance of a vehicle can then be compared to others in a group in order to highlight inefficiencies.

OpenGTS is a pure Java solution running on the Linux, Apache/Tomcat, MySQL and PHP stack. It can also be enhanced to use the proxy services of nginx for SSL. Typically, the OpenGTS software is deployed on a central server, with the client running on mobile devices such as smartphones or automotive computers. As a proof of concept, the OpenGTS server and the python-based sample client will both run on the ODROID-C1. My initial plan was to mount the C1, screen, battery and GPS dongle on a bike, but settled for using it in a car due to lack of time. Note that the C1 is both powerful enough to be used as a central server and inexpensive enough to be in a vehicle. The portable setup is shown in Figure 1.

Figure I: ODROID-CI, GPS, touchscreen and battery setup

Requirements

1. An ODROID-C1, although any ODROID system may be used.
2. Accessories such as an HDMI cable, CAT 5E+ ethernet cable or WIFI 3 dongle, recommended PSU, RTC battery, ODROID-VU 10" touchscreen, or 3.2" LCD touchscreen.
3. A 16GB+ eMMC 5.0 module or 16GB+ Class 10 microSD card, with the latest C1 specific lubuntu desktop image (hard-float), along with an microSD card reader/writer and compatible USB adapter.
4. USB GPS module from Hardkernel, available at http://bit.ly/1EPERhm.
5. A network where the device has access to the Internet and the ODROID forums at http://forum.odroid.com.
6. Oracle Java 8
7. OpenGTS software version: 2.5.8 gpsd & gpsd-clients.
8. Networked access to the C1 via utilities like PuTTY, FileZilla, TightVNC Viewer for MS Windows 7+, ot Terminal for OSX and Linux from a development desktop.
9. LiPo battery packs with a total of two USB ports providing 2A each. One port will be used for the VU if the 3.2" touchscreen is not used, and the other is for the C1 itself. Two USB power cables are also required, which are available from http://bit.ly/1GU9RIS.

Install Lubuntu

To begin, install the latest C1 Lubuntu image onto the eMMC module or microSD card and insert it into the C1. With the ODROID-VU display attached, boot up the system. Run the ODROID Utility and set the display resolution to 720p and reboot. Then, expand the installation partition to use all available space by selecting the Resize your root partition option. Reboot and re-run the ODROID Utility again, to configure and update all remaining aspects of the system, rebooting afterwards. Ensure that you are always logged in as the default "odroid" user, unless otherwise specified.

Update the system

Run the following commands in order to update the Ubuntu packages to the most recent version. The last command fetches fetch the latest C1 kernel, just in case it was held back:

```
$ sudo apt-get autoremove && sudo apt-get update
$ sudo apt-get dist-upgrade && sudo apt-get upgrade
$ sudo apt-get install linux-image-c1
```

After the updates are completed, shutdown the ODROID, attach the rest of the accessories and cables, including the GPS dongle, and reboot again. Once the system boots up, verify the operating system version from a Terminal window using the following command:

```
$ uname -a
Linux c1-1 3.10.73-81 #1 SMP PREEMPT Mon Apr 6 13:17:28 BRT 2015 armv7l
armv7l armv7l GNU/Linux
```

Verify Java version

OpenGTS requires at least Oracle Java version 7, with version 8 being used in this article, which should be installed by default on the official Hardkernel Lubuntu image. This can be verified using either of the following commands:

```
$ which java
/usr/bin/java

$ java -version
java version "1.8.0_33"
Java(TM) SE Runtime Environment (build 1.8.0_33-b05)
Java HotSpot(TM) Client VM (build 25.33-b05, mixed mode)
```

Setup the environment

Setup the JAVA_HOME variable for use by OpenGTS:

```
$ echo "export JAVA_HOME=/usr/lib/jvm/java-8-oracle" >> ~/.bashrc
$ source ~/.bashrc
$ echo $JAVA_HOME
/usr/lib/jvm/java-8-oracle
```

Set the following additional variables, even though their respective components have not be installed yet, just so that all relevant settings are defined in one place:

```
$ echo "export ANT_HOME=/opt/ant" >> ~/.bashrc
$ source ~/.bashrc
$ echo $ANT_HOME
/opt/ant
$ echo "export PATH=$ANT_HOME/bin:$PATH" >> ~/.bashrc
$ source ~/.bashrc
$ echo "export CATALINA_HOME=/opt/tomcat" >> ~/.bashrc
$ source ~/.bashrc
$ echo $CATALINA_HOME
/opt/tomcat
$ echo "export PATH=$CATALINA_HOME/bin:$PATH" >> ~/.bashrc
$ source ~/.bashrc
```

Next, update the PATH variable using a single space after the export command, with the rest on a single line:

```
$ export
PATH=/opt/opengts/bin:/opt/tomcat/bin:/opt/ant/bin:/usr/local/sbin:/usr/
local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games
```

Setup the environment for the administrative user so that the installed software is visible when logged in as root:

```
$ sudo visudo
```

Add the following line and save the file by pressing Control-K and X:

```
Defaults    env_keep += "GTS_HOME"
```

Check the GPS dongle

Obtain the USB information related to the GPS dongle to ensure that it can be used by OpenGTS. In this case, U-Blox is one of the supported vendors:

```
$ lsusb
…
Bus 001 Device 005: ID 1546:01a6 U-Blox AG
…

$ sudo lsusb -D /dev/bus/usb/001/005 | grep "Vendor"
…
  idVendor               0x1546 U-Blox AG
…
```

Check the USB serial port (TTY) where the GPS is mounted, which will be used by the OpenGTS client software:

```
$ sudo ls -lsa /dev/ttyA*
0 crw-rw---- 1 root dialout 166, 0 Dec 31  1979 /dev/ttyACM0
```

You can verify that the GPS dongle is working properly using the following command:

```
$ sudo cat /dev/ttyACM0 | grep GPRMC
$GPRMC,161053.00,A,3719.54074,N,12201.49867,W,0.079,,110415,,,A*65
$GPRMC,161054.00,A,3719.54074,N,12201.49867,W,0.085,,110415,,,A*65
$GPRMC,161055.00,A,3719.54074,N,12201.49867,W,0.024,,110415,,,A*66
```

The Recommended Minimum sentence C (RMC) of the National Marine Electronics Association (NMEA) specification is of interest to us. The RMC - NMEA has its own version of essential GPS PVT data (position, velocity, time) in the following format, which is recognized by OpenGTS:

```
$GPRMC,123519,A,4807.038,N,01131.000,E,022.4,084.4,230394,003.1,W*6A

  RMC               Recommended Minimum sentence C
  123519            Fix taken at 12:35:19 UTC
  A                 Status A=active or V=Void.
  4807.038,N        Latitude 48 deg 07.038' N
  01131.000,E       Longitude 11 deg 31.000' E
  022.4             Speed over the ground in knots
  084.4             Track angle in degrees True
  230394            Date - 23rd of March 1994
  003.1,W           Magnetic Variation
  *6A               The checksum data, which always begins with *
```

Install gpsd

To test the higher level functionality of the GPS dongle, we use gpsd, a service daemon, to monitor one or more GPS signals and deliver the PCV (position, course, velocity) data via TCP port 2947 of the host system:

```
$ sudo apt-get install gpsd gpsd-clients foxtrotgps
$ sudo dpkg-reconfigure gpsd

Q.  Start gpsd automatically?
    <Yes>
Q.  Should gpsd handle attached USB GPS receivers automatically?
    <Yes>
C.  Device the GPS receiver is attached to:
    /dev/ttyACM0
```

```
C.  Options to gpsd:
    <blank>
C.  gpsd control socket path:
    /var/run/gpsd.sock

$ sudo reboot
```

After the device has rebooted, run the cgps minimalist command, then the gpsmon command:

```
$ cgps -s
+-----------------------------------------++-----------------------------------+
|     Time:          2015-04-09T02:20:10.000Z ||PRN:   Elev:  Azim:  SNR:  Used: |
|     Latitude:      37.195407 N              ||   2       59      179     35      Y    |
|     Longitude:    122.014987 W             ||   3 00     031     00      Y    |
|     Altitude:     121.2 m                  ||   6 60     071     32      Y    |
|     Speed:         0.0 kph                  ||  10       15      117     14      Y    |
|     Heading:       0.0 deg (true)           ||  12       47      318     39      Y    |
|     Climb:         0.0 m/min                ||  14       01      317     00      Y    |
|     Status:        3D FIX (133 secs)        ||  17       22      058     16      Y    |
|     Longitude Err:   +/- 11 m               ||  20 14    140     00      N    |
|     Latitude Err:  +/- 13 m                    ||  24    55      250     45      N    |
|     Altitude Err:  +/- 34 m                    ||  25    11      306     32      N    |
|     Course Err:    n/a                        ||  28    03      114     18      N    |
|     Speed Err:     +/- 98 kph                 ||  29    03      253     00      N    |
|     Time offset:   0.078                      ||                                   |
|     Grid Square:   CM87xh                     ||                                   |
+-----------------------------------------++-----------------------------------+

$ gpsmon

localhost:2947:              Generic NMEA>
+-------------------------------------------------------------------------------+
|Time: 2015-04-09T02:21:50.000Z Lat:  37 19' 54.074" N Lon: 122 01' 49.868" W   |
+---------------------------------- Cooked PVT ----------------------------------+
+-------------------------------------------------------------------------------+
| GPRMC GPVTG GPGGA GPGSA GPGSV GPGLL                                           |
+---------------------------------- Sentences --------------------------------+
+------------------++-------------------------------++-----------------------------+
|Ch PRN  Az El S/N ||Time:   022150.00        ||Time:       022150.00          |
| 0   2 177 64  34 ||Latitude:       3719.54074 N   ||Latitude:  3719.54074       |
| 1   5 167  0   0 ||Longitude:  12201.49867 W   ||Longitude: 12201.49867      |
| 2   6  64 58  29 ||Speed:  0.101           ||Altitude:  112.3            |
| 3  10 114 18  11 ||Course:                          ||Quality:   1   Sats: 08         |
| 4  12 321 51  43 ||Status: A       FAA: A  ||HDOP:        1.27                |
| 5  14 313  1   0 ||MagVar:                          ||Geoid:      -30.1             |
| 6  17  61 19  24 |+----------- RMC ------------++----------- GGA ------------+
| 7  20 137 17   0 |+-------------------------------++-----------------------------+
| 8  24 243 53  43 ||Mode: A 3                       ||UTC:         RMS:       |
| 9  25 308 14  18 ||Sats: 24 6 2 12 17 10 25 29 ||MAJ:           MIN:       |
|10  28 117  0   0 ||DOP: H=1.27  V=1.41  P=1.90 ||ORI:           LAT:       |
|11  29 256  6  22 |+----------- GSA ------------+|LON:           ALT:       |
+------ GSV -------+                               +----------- GST ------------+
(68) $GPGSV,3,1,12,02,64,177,34,05,00,167,,06,58,064,29,10,18,114,11*75\x0d\x0a
...
```

Next, launch the foxtrotgps graphical application, as shown in Figure 2:

```
$ foxtrotgps &
```

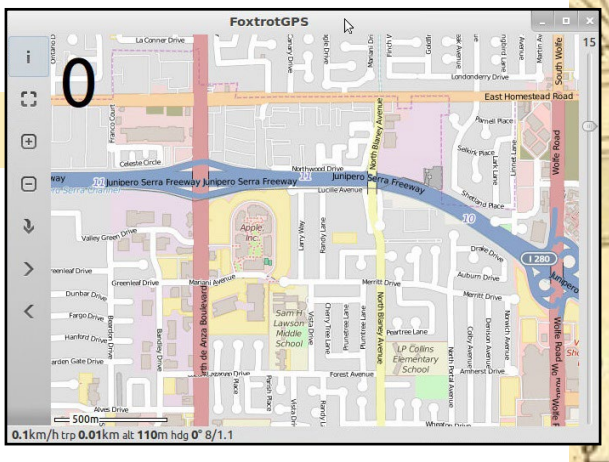


Figure 2: FoxTrotGPS UI

Note that the display of a map requires the presence of a working Internet connection. If you wish to see a real-time map while you are driving, you will need to use your smartphone as a hot-spot and have the ODROID communicate with it via WiFi.

Then, check the list of open files (lsof) dealing with the port 2947 indicated above, which should include foxtrot:

```
$ sudo lsof -i :2947
[sudo] password for odroid:
COMMAND    PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
gpsd       494 nobody   4u  IPv4   2721      0t0  TCP c1-1:gpsd (LISTEN)
gpsd       494 nobody   7u  IPv4  15438      0t0  TCP c1-1:gpsd->c1-1:46077 ESTAB.)
gpsd             1045 nobody  4u  IPv6   2979       0t0  TCP localhost:gpsd (LISTEN)
foxtrotgp  2836 odroid 12u  IPv4  15437      0t0  TCP c1-1:46077->c1-1:gpsd (ESTAB.)
```

Install Javamail

OpenGTS can be setup to send emails to users based on certain events using Javamail, which may require additional configuration:

```
$ mkdir ~/gps && cd ~/gps
$ wget -c http://java.net/projects/javamail/downloads/download/javax.mail.jar
$ sudo cp javax.mail.jar $JAVA_HOME/jre/lib/ext/.
```

Install ant

Ant, which is an open-source build tool, should then be installed:

```
$ cd / && sudo mkdir /opt
$ cd ~/gps
$ wget -c http://apache.mirrors.tds.net/ant/binaries/\
  apache-ant-1.9.4-bin.zip
$ unzip apache-ant-1.9.4-bin.zip && mv apache-ant-1.9.4 ant
$ sudo mv ant/ /opt && cd /opt && sudo chown -R root:root ant/
```

Install Tomcat

Tomcat is a web server that primarily services Java applications, and may be installed using the following commands:

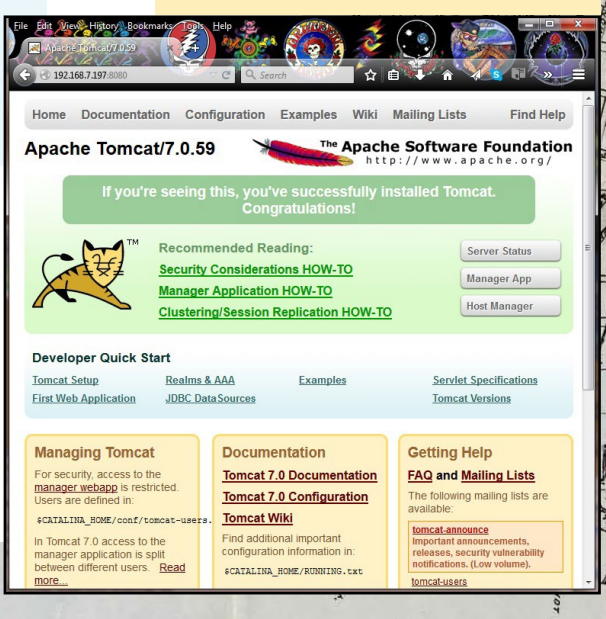
```
$ cd ~/gps
$ wget -c http://apache.mirrors.tds.net/tomcat/tomcat-7/v7.0.59/bin/\
  apache-tomcat-7.0.59.tar.gz
$ tar xvzf apache-tomcat-7.0.59.tar.gz && \
  mv apache-tomcat-7.0.59 tomcat
$ sudo mv tomcat/ /opt && cd /opt && sudo chown -R root:root tomcat/
$ cd /opt/tomcat/conf
$ sudo chmod 644 catalina.*
$ sudo chmod 644 *.xml
$ sudo chmod 644 *.properties
```

You can then verify that Tomcat is up and running at the default HTTP port 8080 after rebooting:

```
$ sudo $CATALINA_HOME/bin/startup.sh
```

Wait for a few moments, then navigate to http://localhost:8080 using the default web browser on the ODROID. You should see the welcome screen shown in Figure 3.

Figure 3: Tomcat welcome page



Install MySQL and JDBC

Install MySQL and its related software using the following commands, setting the root MySQL user password to "odroid" during setup:

```
$ cd ~/gps
$ sudo apt-get install mysql-server mysql-client libmysql-java
```

Next, install the system database and secure its installation:

```
$ sudo mysql_install_db
$ sudo mysql_secure_installation
```

Then, install the MySQL JDBC Driver (J Connector):

```
$ cd ~/gps
$ wget -c http://dev.mysql.com/get/Downloads/Connector-J/\
  mysql-connector-java-5.1.35.zip
$ cd mysql-connector-java-5.1.35/
$ sudo cp mysql-connector-java-5.1.35-bin.jar \
  $JAVA_HOME/jre/lib/ext/.
```

Download and build OpenGTS

Install OpenGTS version 2.8.5 using the following commands:

```
$ cd ~/gps
$ wget -c http://downloads.sourceforge.net/project/opengts/\
  server-base/2.5.8/OpenGTS_2.5.8.zip
$ unzip OpenGTS_2.5.8.zip && mv OpenGTS_2.5.8 opengts
$ sudo mv opengts/ /opt && cd /opt && sudo chown -R root:root opengts/
```

Then, update the webapp.conf file and reboot:

```
$ cd /opt/opengts
$ sed -i "s/#gprmc.logName=.*/gprmc.logName=gprmc/" \
  $GTS_HOME/webapp.conf
$ sed -i "s/#gprmc.parm.account=.*/gprmc.parm.account=acct/" \
  $GTS_HOME/webapp.conf
$ sed -i "s/#gprmc.parm.device=.*/gprmc.parm.device=dev/" \
  $GTS_HOME/webapp.conf
$ sed -i "s/#gprmc.parm.status=.*/gprmc.parm.status=code/" \
  $GTS_HOME/webapp.conf
$ sed -i "s/#gprmc.parm.gprmc=.*/gprmc.parm.gprmc=gprmc/" \
  $GTS_HOME/webapp.conf
```

Next, setup the required symlinks and build OpenGTS:

```
$ sudo ln -s $JAVA_HOME /usr/local/java
$ sudo ln -s $CATALINA_HOME /usr/local/tomcat
$ sudo ln -s $GTS_HOME /usr/local/gts
$ cd $GTS_HOME
$ ant all
```

```
...
BUILD SUCCESSFUL
Total time: 1 minute 39 seconds
```

The OpenGTS service servlets may then be built:

```
$ cd build
$ sudo find .  -name "*.war" | sort -u
./events.war
./gc101.war
./gprmc.war
./gpsmapper.war
./mologogo.war
./track.war

$ sudo find .  -name "*.jar" | sort -u
./lib/aspicore.jar
./lib/astra.jar
./lib/dmtpserv.jar
./lib/gtsdb.jar
./lib/gtsdmtp.jar
./lib/gtsutils.jar
./lib/icare.jar
./lib/lantrix.jar
./lib/sipgear.jar
./lib/taip.jar
./lib/template.jar
./lib/tk10x.jar
./lib/tools.jar
./lib/warmaps.jar
./lib/wartools.jar
```

Please refer to the OpenGTS_config.pdf guide, which is linked in the References section at the end of this article, for details on the options available in the configuration files.

The next step is to deploy the built *.war files to Tomcat:

```
$ sudo cp build/*.war $CATALINA_HOME/webapps/.
```

Whenever configuration files are changed, make sure to run the following command before re-deploying the *.war files:

```
$ ant all
```

Restart Tomcat in order to launch the OpenGTS application:

```
$ sudo $CATALINA_HOME/bin/shutdown.sh
$ sudo $CATALINA_HOME/bin/startup.sh
```

Check the installation using the command:

```
$ bin/checkInstall.sh
Memory-Mb: Max=247.5, Total=13.6, Free=11.1, Used=2.5 [1%]
…
No errors reported

-- Found 2 Warning(s):
1) Memory below recommended value
2) Not fully tested with Java 1.8

-- Recommendations:
- Highly recommend increasing memory to at least 4096 Mb for a production
environment.
```

Update the MySQL database

If OpenGTS fails to create a MySQL server user, using the following commands to do so:

```
$ mysql -u root -p
...
mysql> CREATE USER 'gtsroot'@'localhost' IDENTIFIED BY 'odroid';
mysql> GRANT ALL PRIVILEGES ON * .  * TO 'gtsroot'@'localhost';
mysql> FLUSH PRIVILEGES;
mysql> commit;
mysql> select user,host from mysql.user;
+------------------+-----------+
| user             | host      |
+------------------+-----------+
| root             | 127.0.0.1 |
| root             | ::1       |
| debian-sys-maint | localhost |
| gtsroot          | localhost |
| root             | localhost |
+------------------+-----------+
5 rows in set (0.00 sec)
```

Then, update the database credentials:

```
$ cd /opt/opengts
$ sed -i 's/ServiceAccount.db.name=.*/ServiceAccount.db.name=gts}/' \
  $GTS_HOME/common.conf
$ sed -i 's/ServiceAccount.db.user=.*/ServiceAccount.db.user=gtsroot}/' \
  $GTS_HOME/common.conf
$ sed -i 's/ServiceAccount.db.pass=.*/ServiceAccount.db.pass=odroid}/' \
  $GTS_HOME/common.conf
$ sed -i 's/#db.sql.rootUser=root db.sql.rootUser=root/' \
  $GTS_HOME/common.conf
$ sed -i 's/#db.sql.rootPass=rootpass db.sql.rootPass=odroid/' \
  $GTS_HOME/common.conf
$ cd /opt/opengts
$ sed -i "s/#db.sql.dbname=gts/db.sql.dbname=gts/" \
  $GTS_HOME/config.conf
```

```
$ sed -i "s/#db.sql.user=gts/db.sql.user=gtsroot/" \
  $GTS_HOME/config.conf
$ sed -i "s/#db.sql.password=opengts/db.sql.password=odroid/" \
  $GTS_HOME/config.conf
```

Initialize the OpenGTS database:

```
$ bin/initdb.sh -rootUser=root -rootPass=odroid
Version: 2.5.8
Database created: gts
Privileges granted to user: gtsroot

Validating table columns …
--------------------------------------------------------------------------------
  Table 'Account'                   --- Creating table ...
  Table 'AccountString'             --- Creating table …
  …
  Table 'EventData'                 --- Creating table ...
  …
--------------------------------------------------------------------------------
Column validation completed successfully.
--------------------------------------------------------------------------------

Updating GTS Version: 2.5.8
Updating DMTP Version: 1.3.6
```

The EventData table has all the gprmc (sentence) data obtained from the GPS dongle. The following command loads some sample OpenGTS data:

```
$ cd $GTS_HOME
$ ./sampleData/loadSampleData.sh
...
(Loaded 87 records from file './sampleData/EventData.txt' into table 'EventData')
```

Next, create the sysadmin account for OpenGTS:

```
$ cd $GTS_HOME
$ bin/admin.sh Account -account=sysadmin -pass=odroid -create
Entry Point: org.opengts.db.tables.Account
Created Account-ID: sysadmin
```

Rebuild OpenGTS and deploy changed *.war files, then restart Tomcat:

```
$ cd $GTS_HOME
$ ant all
$ sudo cp build/*.war $CATALINA_HOME/webapps/.
$ sudo $CATALINA_HOME/bin/shutdown.sh
$ sudo $CATALINA_HOME/bin/startup.sh
```

Check to see if OpenGTS is running properly by examining the log files, especially the gprmc log, which will contain an entry every time the OpenGTS client POSTs a gprmc sentence to the OpenGTS server:

```
$ ls -lsa /opt/opengts/logs/*gprmc*
4 -rw-r--r--  1 root   root  351 Apr 11 13:59 w-gprmc.log
```

Configure OpenGTS

Once the entire infrastructure has been built, configured and deployed, navigate to `http://<C1-IP-address>:8080/track/Track` using any browser on the network. A login page will appear as shown in Figure 4.

Figure 4: OpenGTS login page

Based on the configuration setup earlier, enter the following administrative user information, then click on the Login button to view the welcome page, as shown in Figure 5:

```
Account:     sysadmin
Password:    odroid
```

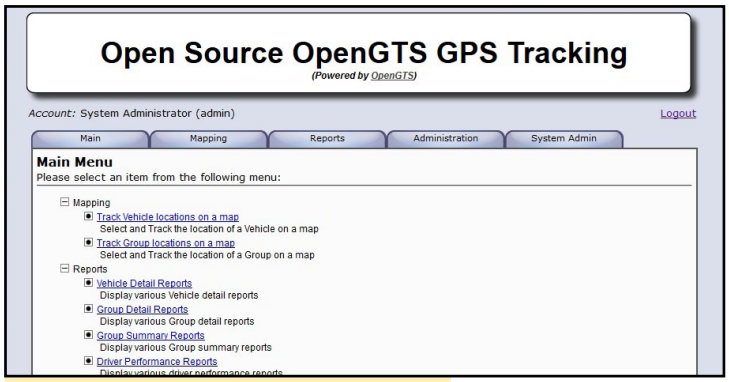


Figure 5: OpenGTS welcome page

Configure the sysadmin user information by selecting that option and updating its data using the steps illustrated in Figures 6 and 7. Then, setup an account for the default "odroid" user as shown in Figure 8.

This user should be given comprehensive privileges so that the system can be managed from that login. The Linux "odroid" user will be used by default so that the OpenGTS client application can POST GPS data to OpenGTS server

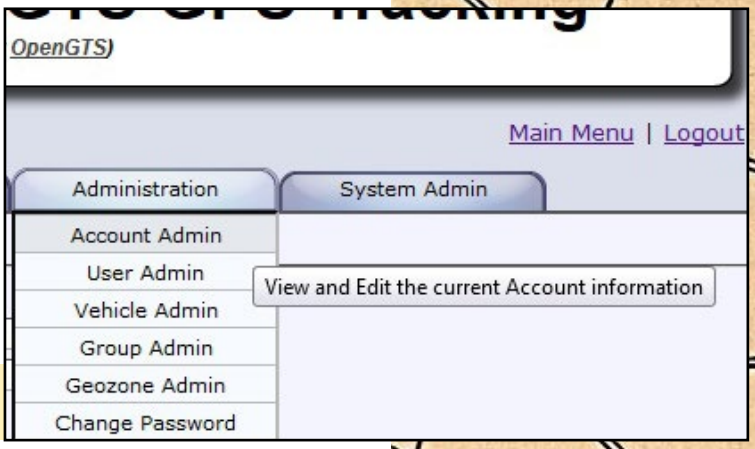


Figure 6: Select administration of admin account

Figure 7: Update admin account

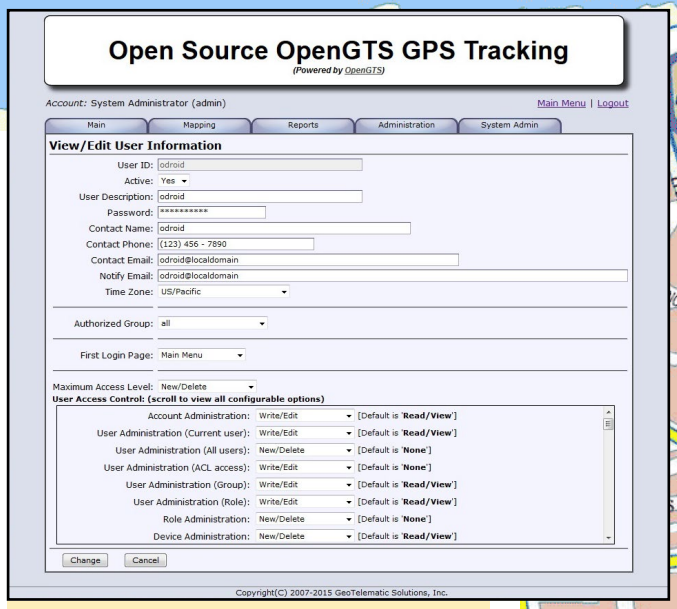


Figure 8: Account setup of odroid user

using the Tomcat servlet. Logout and re-login to the OpenGTS system using the credentials of the newly created "odroid" user. I used a password of "odroid" in this example, to keep things simple. As the "odroid" user, add information for a vehicle you wish to track. In a real situation, one would enter fleets of vehicles. The setup screen is shown in Figure 9.

To relate the vehicle to the embedded ODROID-C1, I used an ID that was derived from the MAC address of the C1. You can use any nomenclature system, as long as the names are unique.

OpenGTS client

To illustrate a basic client, I sent gprmc data via the REST API by navigating to the following address using a web browser:

```
http://<C1's-ip-address>:8080/gprmc/Data?acct=odroid&dev
=001e06a99141&code=0xF020&gprmc=$GPRMC,210549.00,A,3719.
5407,N,12201.4987,W,0.026,,110415,,,A*6B
```

I then used the following python code by creating a script called ~/gps/gps-rest-client.py that reads the TTY port where the GPS is mounted, gets the gprmc sentence, then sends it off to the OpenGTS server:

```
#!/usr/bin/python

#
# gps-rest-client: Util to POST gps rmc info
#                  to the opengts servlet.  Presumes
#                  gps receiver works off ttyACM0
#
# Venkat Bommakanti
# 04/11/15
```

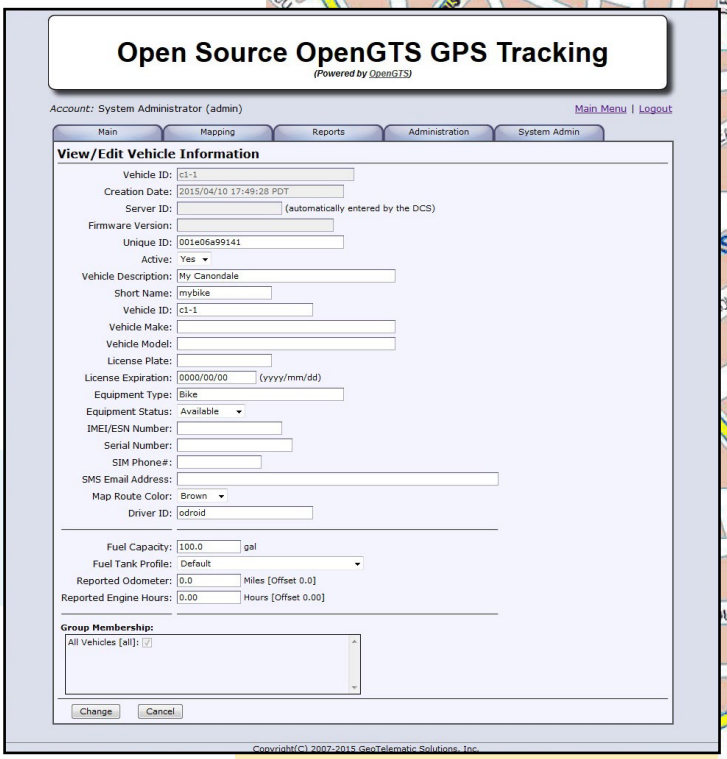


Figure 9: Setup of vehicle to be tracked

```
#
# Free to use at your own risk.  No warranties implied.
#

import sys, time
import httplib, urllib

# setup http header to be used by client
headers = {
  "Content-type": "application/x-www-form-urlencoded",
  "Accept":          "text/plain"
}

# initialize
close_ok = True

try:
  while True:
    # open device to read gps data from receiver
    if (close_ok):
      sys.stdin = open('/dev/ttyACM0', 'r')
      close_ok = False
      print "Opened tty"

    gps_line = None

    # warning: no timeout, could hang
    gps_line = sys.stdin.readline()
    print '.'

    # look for line with '$GPRMC
    if (gps_line.find('$GPRMC') >= 0):
      print '\n' + time.strftime('%X') + ': '  + gps_line

      # prep REST params according to user/vehicle setup
      params = urllib.urlencode({
        'acct':    'odroid',
       'dev':     '001e06a99141',
       'code':    '0xF020',
       'gprmc':    gps_line.strip('\n'),
      })

      # create HTTP connection to servlet running
      # "local" to this script.  Use the C1's ip-address
      # if available in your mobile setup.
      conn = httplib.HTTPConnection("localhost:8080")
      conn.request("POST", "/gprmc/Data", params, headers)

      # should get OK response
      response = conn.getresponse()
      print response.status, response.reason
```

```
        data = response.read()
        print data

        # done reading for now
        conn.close()

        # close tty read
        sys.stdin.close()
        close_ok = True
        print "Closed tty"

        # repeat search every 60 secs
        print "Waiting 2 mins...\n"
        time.sleep(120)

    # go to read more of gps data...
    # while block ends
except:
    print "\nExiting...\n"

print "Done."
```

This example script opens the TTY port every 2 minutes, fetches the data, sends it via a POST request, then closes the port, which is sufficient to prove the concept. Make this file executable by changing its file permissions:

```
$ chmod 755 gps-rest-client.py
```

Many open-source GPS clients exist that actually use the services offered by gpsd. You can research them and chose an appropriate one, or build your own version. By using the gpsd services, you can program against abstract layers without needing to deal with raw port information, making your code portable.

Test the setup

Now that we have a fully functional system, let's try it out! Launch tomcat in a Terminal window:

```
$ cd ~/gps/
$ sudo $CATALINA_HOME/bin/shutdown.sh
$ sudo $CATALINA_HOME/bin/startup.sh
```

Then, run the previously mentioned Python script in a separate Terminal window:

```
$ python ./gps-rest-client.py
```

Drive around for a while in the vehicle, the go back home. Shutdown the system, re-attach to the network with a working Internet connection, then launch and login to the OpenGTS web interface at http://<C1-IP-address>:8080/track/Track. On the welcome page, select the "Track Vehicle Locations On A Map" link, as shown in Figure 10. You can then explore the functionality and power of OpenGTS according to the standard guides available for OpenGTS.

Disclaimer

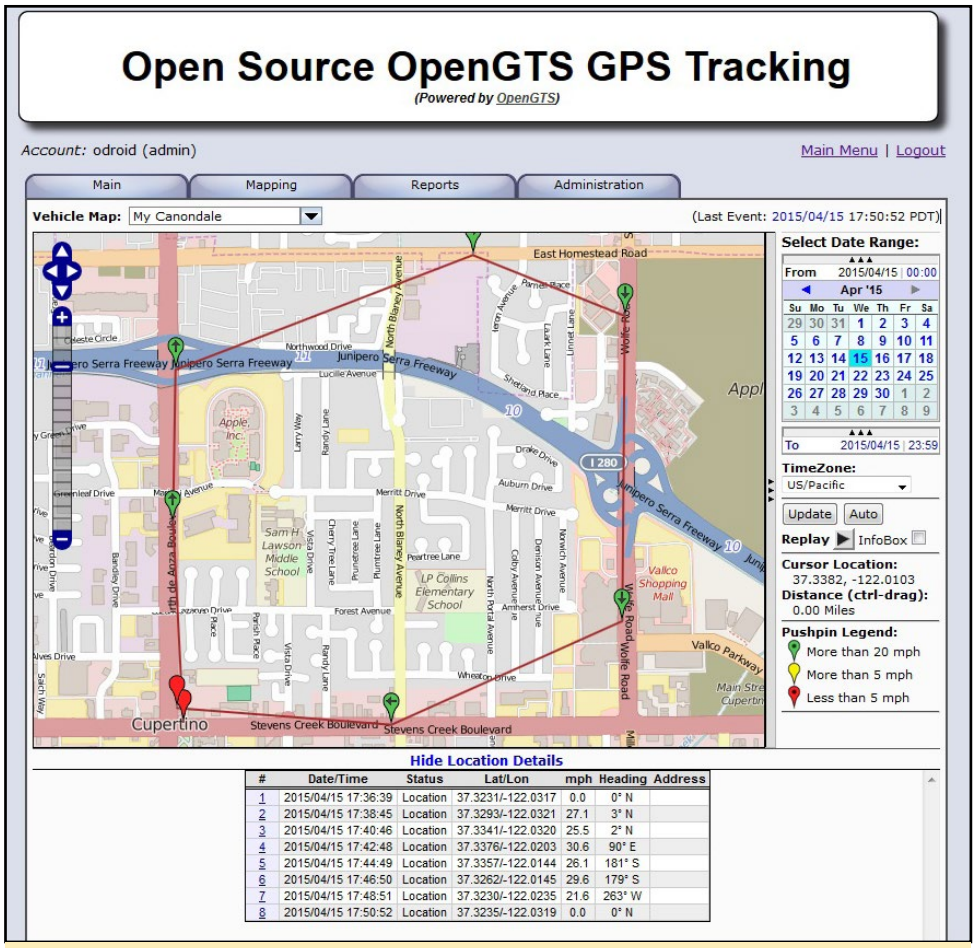


Figure 10: See the ODROID-C1 go around a famous "fruit" company

Be cautious when manipulating your configuration inside a vehicle, and ensure that safety comes first before attempting any activity on the setup. HardKernel, the contributors of these articles and forum members cannot be held liable for possible mishaps during your experiments.

OpenGTS Resources

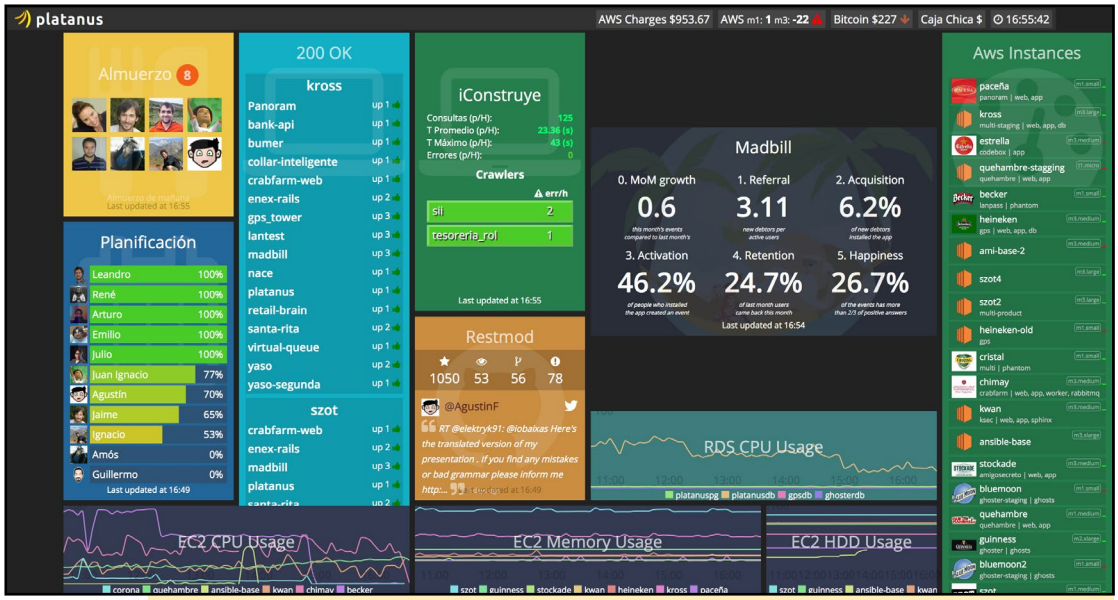
For additional information or questions, please refer to the following links:

```
http://bit.ly/1GUtTcA
http://bit.ly/1KFU6tX
http://bit.ly/1I6UCRV
http://bit.ly/1KFU7hB
http://bit.ly/1QdEr8f
http://bit.ly/1iy4LLu
http://bit.ly/1KFUbO8
http://bit.ly/1DYbJAy
http://bit.ly/1EQdpA5
http://bit.ly/1QdEvF0
http://bit.ly/1GK8H3E
```

USE AN ODROID-C1 AS A WALL DASHBOARD

KEEP TRACK OF YOUR ENTERPRISE PROJECTS IN REAL TIME ON A LARGE SCREEN

by @platanus



Goodbye whiteboards with colored markers - the ODROID-C1 dynamic dashboard is here!

At Platanus (www.platan.us), we have a 42" monitor that displays some metrics about our products and resources along with some fun and useful information. The software we are using behind it is called Dashing (dashing.io), which is a really cool Sinatra-based application that can create dashboards containing different information widgets.

I originally wrote this article as a guide for getting the dashboard working on a Raspberry Pi, but the problem was that I wanted to run Chromium in order to have all the features of a modern browser. The Raspberry Pi wasn't powerful enough, and the dashboard was unstable and very slow.

As a second alternative to the Pi, I tried an Android stick. The advantage was that you could find dual- or quad-core processors that ran Android with full hardware acceleration. I may have picked the wrong one, but in the end, the kernel was limited to 720p, and our dashboard runs at 1080p, which looked awful. Also, Android is not as nearly tweakable as a bare Linux OS.

Finally, I tried the brand new ODROID-C1, which is small and cheap (US$35, plus all the necessary accessories), with a quad-core processor and 1GB RAM. Hardkernel, the company who makes the ODROID, provides a special version of Ubuntu 14.04 that runs very smoothly, and is completely hardware accelerated. For now, the ODROID-C1 is working very well. This article details the steps that I took in order to have a desktop-free, full-screen dashboard that automatically starts on boot.

To create your own dashboard, you will need the following components:

ODROID-C1 with an Ubuntu SD card or eMMC module
WiFi stick (optional)
Large HDMI monitor

Connect your ODROID to the screen and to the network. You can use a keyboard and mouse if you have one, but everything can be done via an SSH connection.

Resize the partition

Once you've flashed the image onto the SD card or eMMC module, you'll get a 4.6GB partition, regardless of the size of the drive. To do so, run the ODROID Utility script and choose the Resize Partition option:

```
$ sudo ODROID-utility.sh
```

Setup NTP

The next step is to automatically sync the time with the Ubuntu NTP server:

```
$ sudo apt-get install ntpdate
$ sudo ntpdate -u ntp.ubuntu.com
```

```
$ echo "America/Santiago" | sudo tee /etc/timezone
$ sudo dpkg-reconfigure --frontend noninteractive tz-data
```

Update the OS

```
$ sudo apt-get update
$ sudo apt-get -y upgrade
```

Change the device name

```
$ echo "dashboard" | sudo tee /etc/hostname
$ echo "127.0.0.1 dashboard" | sudo tee -a /etc/hosts
```

Disable lightdm

We want to boot directly to our dashboard, which is done by disabling the default desktop manager with the following command:

```
$ echo "manual" | sudo tee -a /etc/init/lightdm.override
```

Configure the monitor

Uncomment the resolution that you want to use by editing the /media/boot/boot.ini file:

```
# setenv m "vga"          # VGA 640x480
# setenv m "480p"         # 480p 720x480
# setenv m "576p"         # 576p 720x576
# setenv m "800x480p60hz" # WVGA 800x480
# setenv m "svga"         # Super VGA 800x600
# setenv m "xga"          # XGA 1024x768
# setenv m "720p"         # 720p 1280x720
# setenv m "800p"         # 800p(WXGA) 1280x800
# setenv m "sxga"         # SXGA 1280x1024
setenv m "1080p"          # 1080P 1920x1080
# setenv m "1920x1200"    # 1920x1200
```

You may have problems with the overscan, which causes the desktop to appear cropped. The best solution is to disable overscan in the TV. Check the display menu options for an overscan item, which will most likely be labeled as one of the following:

```
Overscan
Just scan
Screen fit
HD size
Full pixel
Unscaled
Dot by dot
Native
1:1
```

Configure the network

Run this command to get the name of your network interfaces. With my ODROID, I found that eth1 corresponded to the wired ethernet port, and wlan2 corresponded to my wifi USB stick:

```
$ ifconfig
```

Using ethernet

If you don't have a WiFi stick, you will need to plug the ODROID into a router or cable modem using an ethernet cable. To configure ethernet, add the following to the /etc/network/interfaces.d/eth1 file:

```
$ auto eth1
$ iface eth1 inet dhcp
```

Using wifi

If you have one, plug in your wifi stick and run the following command to list the network interfaces in order to check whether it was detected. Look for an interface named either "wlan0", "wlan1" or "wlan2":

```
$ ifconfig
```

Next, update the network configuration to use the DHCP service by selecting the network and password. Create a file name that corresponds to the network interface name, which is wlan2 in this example, in the `/etc/network/interfaces.d` directory, and add the following code at the end:

```
auto wlan2

allow-hotplug wlan2

iface wlan2 inet dhcp

wpa-ssid "ssid"

wpa-psk "password"
```

Finally, remove the network manager so that the network interfaces may be configured from the command line, instead

of using the Ubuntu desktop:

```
$ sudo apt-get remove network-manager
```

Auto-start the browser

First, you'll need to install Chromium:

```
$ sudo apt-get install -y chromium-browser
```

Then, configure Chromium so it starts maximized to eliminate the window border and use the entire TV screen. Open ~/.config/chromium/default/preferences and edit the following section:

```
...
"browser": {
  ...,
  "window_placement": {
     "bottom": 1080,
     "left": 0,
     "maximized": true,
     "right": 1920,
     "top": 0,
     "work_area_bottom": 1080,
     "work_area_left": 0,
     "work_area_right": 1920,
     "work_area_top": 0
  }
  ...
}
```

X server

Install the X11 server utilities for controlling video parameters, as well as the unclutter application, which removes the mouse cursor from the dashboard:

```
$ sudo apt-get install -y x11-xserver-utils unclutter
```

Create a script file called `/home/odroid/dashboard/autostart.sh`, which runs Chromium in kiosk mode:

```
#!/bin/sh

chromium-browser \
--kiosk \
--disable-restore-session-state \
--start-maximized \
--incognito \
http://dash.platan.us
```

Then, add execution permissions to the script:

```
$ chmod +x /home/ODROID/dashboard
```

Add this code to your `~/.xinitrc` to run unclutter and disable the screensaver:

```
unclutter &

xset s off          # don't activate screensaver

xset -dpms          # disable DPMS (Energy Star) features
xset s noblank      # don't blank the video device

exec /home/odroid/dashboard/autostart.sh
```

To start the dashboard on boot, we need to create an init script in /etc/init.d/dashboard:

```
$ sudo touch /etc/init.d/dashboard

$ sudo chmod 755 /etc/init.d/dashboard
```

Next, add this code to the /etc/init.d/dashboard script:

```
#! /bin/sh
# /etc/init.d/dashboard
case "$1" in
  start)
    echo "Starting dashboard"
    # run application you want to start
    /bin/su odroid -c xinit
    ;;
  stop)
    echo "Stopping dashboard"
    # kill application you want to stop
    killall xinit
    ;;
  *)
    echo "Usage: /etc/init.d/dashboard {start|stop}"
    exit 1
    ;;
esac
exit 0
```

Register the script to start on boot:

```
$ sudo update-rc.d dashboard defaults
```

Then, allow the user to run an X application by editing the

file `/etc/X11/Xwrapper.config` and changing the follow line:

```
allowed_users=anybody
```

Troubleshooting

If your ethernet connection is slow, there may be a problem using a 1000 link with some switches. The ODROID kernel v1.2 defaults to 100Mbps, but you can limit this in the 1.1 kernel by adding a line to the `/etc/rc.local` file:

```
ethtool -s eth0 speed 100 duplex full autoneg off
```

References

- http://bit.ly/1hD2dIn
- http://bit.ly/1Jtsfzn
- http://bit.ly/1GxfCm9
- http://bit.ly/1PHCfpn
- http://bit.ly/1EuxrzR
- https://github.com/Pulse-Eight/libcec

If you have questions or comments, please visit the original article at `http://bit.ly/1GvWSjV`.

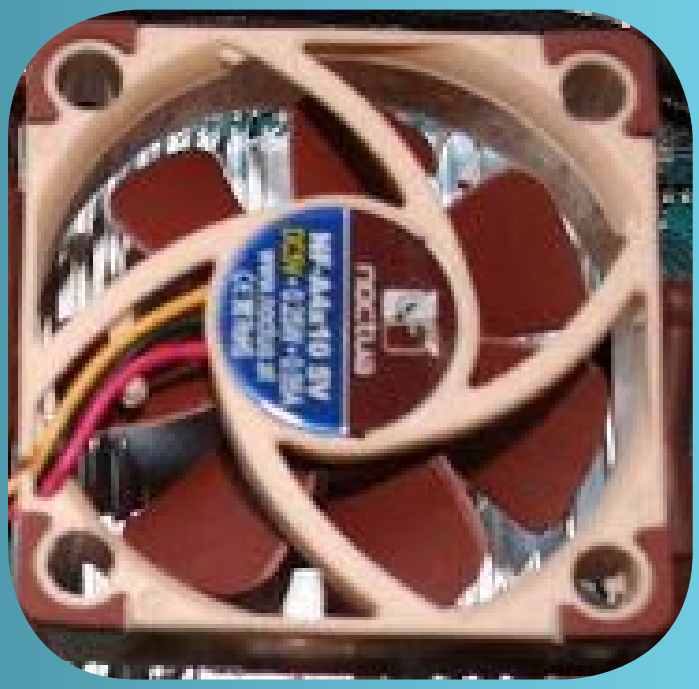
With an ODROID wall dashboard, you can invite a few friends over and recreate your favorite scenes from the movie Wargames, where we learned that the only winning move in thermonuclear war is not to play!

"After careful consideration of all 437 charts, graphs, and metrics, I've decided to throw up my hands, hit the liquor store, and get snockered. Who's with me?!"

IMPROVE YOUR ODROID-XU3 FAN

GO EVEN QUIETER

by Tomasz Nazar



A Noctua NF-A4 cooler fan (`http://bit.ly/1EyrZYP`) runs at a much lower decibel level than the stock XU3 fan. It operates at 5V and takes about 20 minutes to install. To install it, remove the original fan by unscrewing the four holding screws at each corner, and unplugging it from the XU3. Cut the fan connector, then connect it to the new fan with the supplied scotch connectors. Refer to the Noctua installation guide at `http://bit.ly/19af3j4` for more details. Then, plug in the new fan to the XU3, pushing the fan connector gently, but firmly, back onto the fan socket to ensure a solid connection.

Because the original screws are too small to secure the new fan, and the Noctua screws are too big, the best solution is to use 2 of the noise dampener inserts that come with the Noctua fan in order to keep the fan from moving. Close the XU3 case, which secures the fan firmly into position, and enjoy your new fan! When used in conjunction with the improved XU3 fan service described above, the noise level can be lowered to around 18dB.

For more information, or to post questions and comments, please visit the original post at `http://bit.ly/1BeKEqw`.

RETRO GAMING CONSOLE

PUT SOME NEW LIFE IN YOUR OLD CONSOLE CASE

by Jason Ellingsworth

A slim PlayStation 2 case can be repurposed as a slick-looking emulation station

In March of 2015, my sister was sitting next to me at my parent's house while I was on my laptop browsing Youtube videos, when she told me to look up "Pi computers". I didn't really know what to expect, but what I found was a great community of smart people doing amazing things with this fully functioning computer, that also happens to be no bigger than a credit card. I saw robots, helicopters that maintained a specified altitude, and something that interested me greatly: a retro gaming console the size of a pack of cigarettes hooked up to a 1080p TV playing all the old favorites.

I wasn't all that impressed with the Pi's emulation of Atari, NES, SNES, and Genesis games. However, I came across a video of someone playing Mario Kart on a Nintendo 64 emulator, and another video of someone playing Final Fantasy 7, and I became very excited. I did not just want to jump right on to the Raspberry Pi site and buy it without considering alternatives, so I did some research.

I found a great list of amazing little computers at `http://bit.ly/1zhp1vI`. One was the BeagleBone, which was a step up from the original Raspberry Pi, another was the Banana Pi that offered some upgraded specs, and a third was called Minnowboard that had a hefty $130 USD price and the option to run a Windows OS. The article also featured a company named Hardkernel who made a family of ODROID boards that came with a variety of specs.

I was originally going to get a Minnowboard, since I thought that I needed Windows on my device if I was going to get any joy from it. After really thinking about it, I had to decide if what I was going to be able to actually do with this board was worth the $130 USD, and decided to go with something else. With my new knowledge of what was actually out there, and weighing the cost vs. the specs, I decided ODROID was the way to go. They offer an entry-level board competitively priced with the Raspberry Pi 2 at $35 USD, and the ODROID has a lot more power for the cost.

I purchased my parts from an American distributer called Ameridroid (`www.ameridroid.com`). For $95 USD, I got the ODROID-C1, Micro HDMI and power cable, a 16gb MicroSD card preloaded with a lightweight version of Ubuntu called Lubuntu, a WiFi dongle, and a heat sink for the processor. It got to my house in a couple of days, and was a pleasant surprise on the dining room table when I got home from work. I got it all plugged in, and it looked like a mess, with wires coming out everywhere on this little credit card sized board.

After getting it booted up, and experimenting with an operating system that I was not used to at all, I tried to get Emulation Station and RetroArch installed, which are two of the programs I wanted to use in order to emulate retro consoles. RetroArch does the emulation for most consoles, and Emulation Station is a front-end interface for selecting the games in a nice looking user format. I found out, after some frustrating experiences, that getting these applications set up through Ubuntu was for extreme technophiles, and I was in over my head. After asking numerous questions on the ODROID forums, I decided that using the Android operating system was

going to be more suitable to my expertise level, and I could accomplish what I wanted with minimal effort.

After flashing my 16GB card with the most recent version of Android, I got the device booted up, and was presented with a much more familiar interface. The first thing that I noticed missing was my WiFi connection. The dongle that I bought that was advertised as compatible with the Android OS, but it did not initially work. However, with the recent release of the 1.4 version of Android, the WiFi dongle now works perfectly.

While I waited for the WiFi dongle to be supported, I used an ethernet cable instead. I then noticed that the Google Play store was absent from the image. Luckily, it was an easy fix to download the installer from http://bit.ly/1wHG45b, save it to a thumb drive, and plug it into my board. It was a nice, easy one-step install.

However, I uncovered another problem. Many of the games from the Play Store require WiFi to download or run, and do not recognize the fact that I had an ethernet connection. I solved this by downloading a program to fake my WiFi connection from `http://bit.ly/1dmONVe`, although the application is no longer necessary if you use one of Hardkernel's WiFi adapters. After all that, I was pretty much set, although the controller, mouse and keyboard support on many Google Play games varies from game to game. Now all that was left was to deal with the mess of wires on my dresser. I wanted to get the ODROID inside some form of case before somehow damaging it.

I thought about many different options for building a case. I considered Legos, cardboard, and a top-loading NES, then decided that a Playstation 2 (PS2) slim console would be the perfect size for what I needed. At 6" x 9" x 1.1", it is portable enough to carry around in a book bag.

A few searches on Ebay yielded a non-working slim PS2 for $18 shipped. After pulling out the internals, I put them aside to resell on Ebay in order to get some money back. I used a dremel tool to do some light surgery, and ended up with the case shown here.

Most of what needed to be removed had to do with the inside of the DVD tray, which allowed easy access to the internal USB ports. After considering the placement and cost of extension cables to bring the various ports to the exterior of the case, I decided to move the board to the back side of the case so that I could plug the power, OTG USB, and HDMI connections directly into the board. This had the added bonus of allowing more room for the 4-port hub and its connected devices. My plan was to have the main board, 4 port hub, keyboard, mouse, WiFi, and a 128gb flash drive stored inside the case, as well as 4 USB ports mounted to the front of the case for gamepad input and other peripherals.

From top to bottom, all of the steps required to create the modded PS2 case: mounting the USB hub inside it, attaching all components, sealing the components into the back and organizing them for easy access

The next step was to finish cutting up and patching the case. I used some of the plastic casing from an old black light to make a filler plate for the front and back of the PS2 case. As shown in Figure 5, I used a dremel to make holes for the 4 USB ports I brought out to the front of the case, with 2 coming from the ODROID itself, and the other 2 coming from the powered USB hub.

The ODROID board has the 2 ports extended to the front, leaving 1 USB port going to the 4 port hub, and the last one for the WiFi module, which left 2 internal ports free, which I used for a flash drive and a wireless keyboard/mouse dongle. This left 4 open ports in the front for controllers or whatever else I wanted to plug in.

You can't really tell from the pictures, but there are two power cords going into the case. One goes into the power plug of the ODROID, and a second cord plugs into the side into the 4 port hub. This ensures that I can not overdraw power from having too many USB devices connected. These cords combine into a double pole rocker switch which cuts power to both devices after I have executed a shutdown through the OS.

I spent approximately $150 USD total on the project, with some optional components. If anyone else would like to try the same thing, I would love to see how it turns out. I am sure some things could have been done better, but I took the route that required the least amount of knowledge in electronics. I posted a couple videos at `http://bit.ly/1dmQAti` and `http://bit.ly/1ELwEwB` to show the capabilities of the console while running Android and PlayStation Portable games, including Grand Theft Auto and Asphalt Airborne. It has been a lot of fun, and if you are interested in doing something like this, don't be scared! If I can do it, you can too. Enjoy!

If you have any questions or comments, please refer to the original article at http://bit.ly/1HM0xMf.

QUAKE II

THE GAME THAT REVOLUTIONIZED THE FIRST PERSON SHOOTER GENRE

by Jose Cerrejon

The first time that I played Quake I was in 1998. My computer was a PC was an Intel DX4 with a 100Mhz processor and a brand new 3dfx Banshee graphics card. The game represented a huge advance in graphics, and used polygons for the first time. Now I am able to play the sequel to the original game on my ODROID-C1, thanks to a port made by Tobias Schaaf (@meveric). This article will guide you through setting up Yamagi Quake II, which is an enhanced client for Id Software's Quake II that includes some interesting features:

- **OGG music support**
- **High texture packs**
- **Support for unlimited screen size/resolutions**
- **Compatibility with most mods**
- **Multiplayer**

Prerequisites

Quake II runs well using the official Hardkernel v1.3 release of Ubuntu 14.04.02 LTS for ODROID-C1. After copying the image to an SD card or eMMC modules, it is recommended to expand the partition and update the distribution using the ODROID Utility.

To expand the partition, click on the ODROID Utility shortcut on the desktop, then select the "Resize your root partition" option, then reboot after it completes. Type "`df -h`" command in the Terminal window to verify that the entire disk space is being used. If it's not, run the utility again the re-select the resize option.

To update the operating system, type the following into a Terminal window:

```
$ sudo apt-get update && sudo apt-get dist-upgrade -y
&& sudo apt-get autoremove -y && sudo apt-get clean
&& sudo apt-get autoclean
```

Download Quake

Quake II needs some extra files that can be downloaded with any browser:

Quake II Shareware pak files (q2-314-demo-x86.exe):
`http://bit.ly/1JCh4V3`

OGG Extra soundtrack (optional):
`http://bit.ly/1QMCMb5`

You can choose one of the following two methods to install Quake II:

Method 1 - Repository installation

@meveric is a well-known user on the ODROID community forums, and maintains a repository of dozens of applications and games that he has ported to the ODROID. You can add the repository by typing the following into a Terminal window:

```
$ sudo wget -P /etc/apt/sources.list.d \
  http://oph.mdrjr.net/meveric/sources.lists/meveric-
all-testing.list
$ sudo wget -O- \
  http://oph.mdrjr.net/meveric/meveric.asc | \
  sudo apt-key add -
$ sudo apt-get update
```

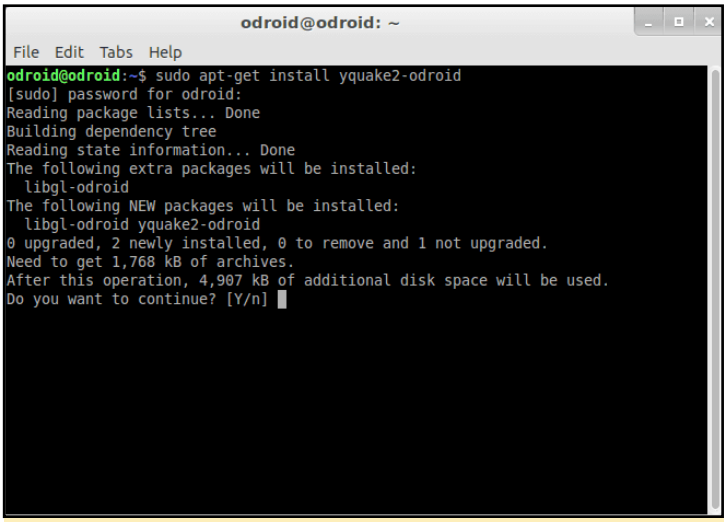


Figure 1 - Installing Quake II

Install yQuake II from Meveric's repository with the following command:

```
$ sudo apt-get install -y yquake2-odroid
```

Next, copy the Quake II data files to ~/.yq2/baseq2, which are available from the previously downloaded q2-314-demo-x86.exe:

```
$ mkdir -p ~/.yq2/baseq2
$ unzip -j q2-314-demo-x86.exe "*.pak" -d ~/.yq2/ba-
seq2
```

The last step is to copy the file game.so to ~/.yq2/baseq2:

```
$ cp /usr/local/share/yquake2/baseq2/game.so ~/.yq2/
baseq2
```

Optionally, you can copy the OGG files from the soundtrack to the folder `~/.yq2/baseq2/music`.

Method 2 - PiKISS installation

PiKISS is a project that I created in order to help the Raspberry Pi community. I'm currently adapting the scripts for compatibility with the ODROID-C1. Basically, it's a menu and set of scripts that allow you to install several useful apps as well as automatically and easily configure your device, including servers, games, emulators, and operating system tweaks.

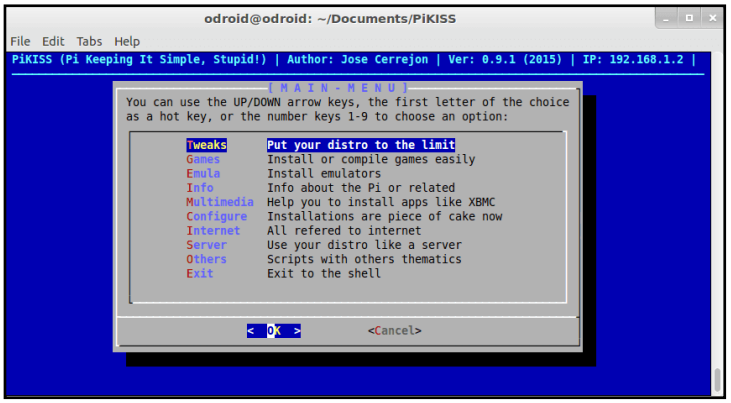


Figure 3 - PiKiss main menu

To install and run PiKiss, type the following into a Terminal window:

```
$ git clone https://github.com/jmcerrejon/PiKISS.git
$ cd PiKISS
$ ./piKiss.sh
```

Once the PiKISS menu appears, choose Games -> Quake. When asked for the administrator password, type "odroid" and choose the Shareware version if you don't have

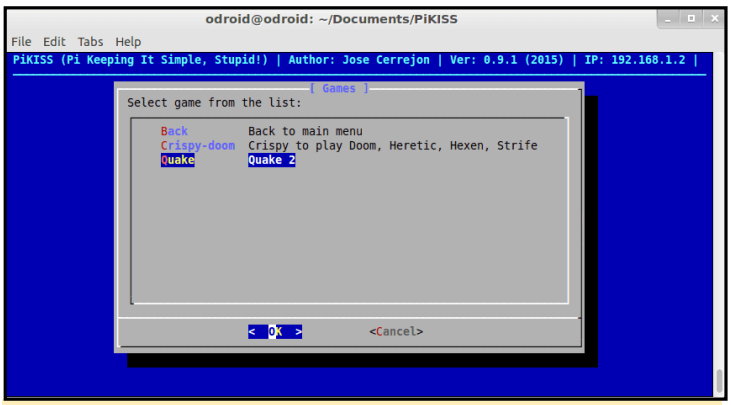


Figure 5 - PiKISS Shareware

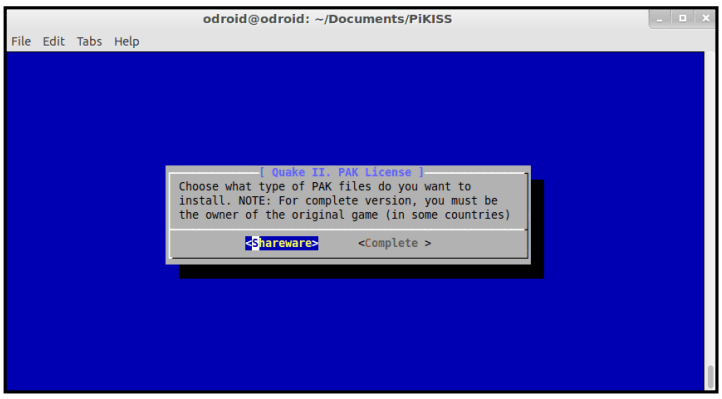


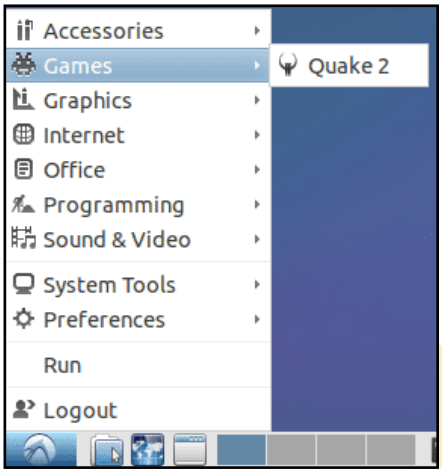
Figure 6 - Quake II Options Menu

the original copy of the game.

The script updates your operating system with @meveric's repositories, installs the yQuake2 packages, then copies all the necessary files to run the game, including the OGG soundtracks, then sets the resolution to 1280x720.

Running Quake II

To play the game, click on the Applications menu and select Games -> Quake 2. For more information visit the PiKISS GitHub project at http://bit.ly/1Es9GqY.



Select here to frag some enemies!

Graphics drivers

If you are running Ubuntu v.1.3 on an ODROID-C1, you may get a Segmentation Fault error when using the r4p0 GPU Mali drivers, which can be fixed by typing the following into a Terminal window:

```
$ cd /usr/lib/arm-linux-gnueabihf/
$ sudo ln -sf libEGL.so.1 libEGL.so
```

Adjust the resolution

If you want to run the game without black borders around the edge, edit the file /home/odroid/.yq2/baseq2/config.cfg. Make sure that you set the parameters gl_customheight and gl_customwidth to correspond to the desired resolution, then set the gl_mode to "-1":

```
set gl_customheight "720"
set gl_customwidth "1280"
...
gl_mode "-1"
```

If you get any other errors, post your issue on the ODROID forums at http://bit.ly/1E66Tm6 in the Games and Emulators section, and I will respond under the username @ulysess.

Gameplay tips

Open the Quake II console by pressing the ` key, which is just to the left of the ''1" button on the keyboard, and type any of the following to enable cheats:

give body armor	I body armor
dmflags 8192	Turns on Infinite Ammo
noclip	Walk Through Walls/Fly
give all	All Items in Single Player
god	God Mode

Quake II is a classic and enduring game, and Id Software deserves thanks for releasing their source code so that everyone may enjoy it on the operating system of their choice.

In Quake II, the goal is to kill all the things!

ARM SOLAR CHALLENGE

JOIN THE RACE TO CREATE A FUNCTIONAL SOLAR-POWERED MICRO-DATA CENTER

by Inveneo

Inveneo, in partnership with ARM Limited, will launch a solar powered Micro-Data Center Design Challenge, starting on March 11, 2015. The top prize for the competition is $10,000, and the winning design will be built and deployed in the developing world.

Inveneo is seeking students, engineers, researchers, and innovators to submit their design of a solar powered micro-data center. Given the harsh environments present in much of the developing world, designers will need to create a functional micro-data center that can be powered with a solar photovoltaic system, withstand intense heat and humidity, and run completely without access to standard air conditioning.

Candidates will use ARM-based solutions to create a "micro-board chassis" design that will use off-the-shelf ARM processor micro boards, such as the Raspberry Pi, Banana Pi/Pro, and ODROID. Inveneo has partnered with LeMaker, which is offering a discounted 15 Banana Pro kit that can be used to build a prototype micro-board chassis.

"We envision a new type of blade server enclosure design. The design will use 15 of these new generation microcomputer boards and will be very low energy usage, DC-powered, and passively cooled," says Bruce Baikie, Executive Director of Inveneo. "Just as BackBlaze changed the low end storage market with their open source design, we are planning to revolutionize the low end blade server market with this challenge."

The contest is open to applicants who are at least 18 years of age, in teams that range from three to seven members. The panel of judges includes industry experts from Inveneo, ARM, and LeMaker, among others. The top two winning designs will be announced on July 15, 2015. If you are interested in entering this design challenge or to find more information, please visit `http://bit.ly/1Fewri2`.

Inveneo is a San Francisco-based 501(c)(3) non-profit social enterprise that designs and delivers sustainable computing, as well as better access to broadband Internet to those who need it most in the developing world. Inveneo enables organizations working in developing areas to better serve people in need. The team works to transform lives through access to education, healthcare, economic opportunity and relief. Inveneo and its partners have delivered projects in 31 countries and are impacting the lives of over 3 million people in some of the poorest and most challenging international regions.

ELECTRONIC SUPERPOWERS

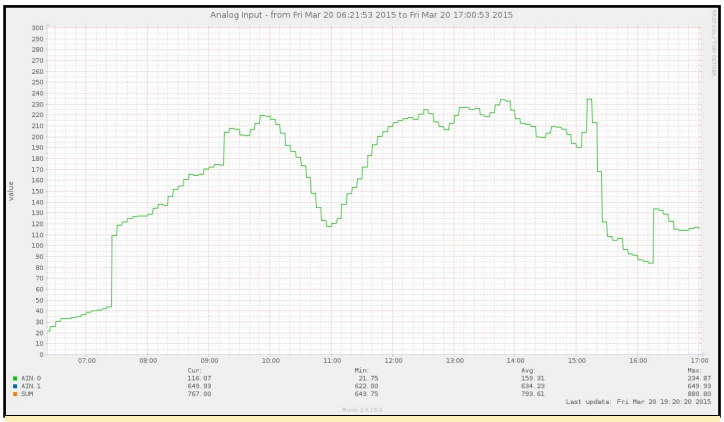
OBSERVING A SOLAR ECLIPSE ON A CLOUDY DAY

by Pascal Pucholt

Solar eclipses have always fascinated human beings. They are so impressive that, for a long time, they were dreaded as a sign of evil coming over the earth. But in modern times, experiencing a solar eclipse, even a partial one, is a spectacular experience. On March 20th 2015, Svalbard and the Faroe Islands experienced a total solar eclipse, and in many areas of Europe, the sun was partially covered by the moon. I live in Uppsala, Sweden, where the sun was covered up to 85.8% at the apex. I had previously experienced a total solar eclipse in 1999 and was looking forward to this event, since it was such a strong shading. However, when the day of the solar eclipse arrived, a thick cover of clouds was hiding the sun, and only with a lot of imagination could the solar eclipse be perceived directly, as shown in Figure 1.

Although difficult to detect with the human eye, a solar eclipse that is obscured by clouds can still be seen by an ODROID

I was rather disappointed, and thought that I had missed out on the whole thing. However, I had an ODROID-C1 sitting at home on the window sill, which was hooked up to a photo resistor on the analog input port. When I checked the recorded light intensity values of my little ODROID, I found a nice surprise! As seen in Figure 2, the light sensor detected the change in light intensity of the eclipse, and the munin daemon running on the computer saved it to the database for me.



Changes in light intensity as recorded by the light sensor coupled to the analog input of my ODROID-CI

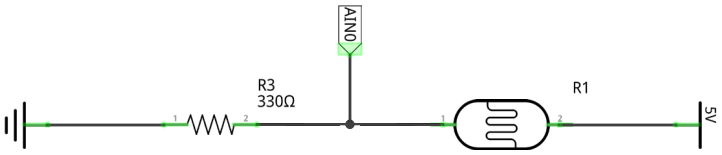
The first contact between the sun and moon started at 9:52AM, the maximum coverage was reached at 11:00AM in Uppsala, and by 12:08PM, the eclipse ended. The times could easily read directly from the light intensity record, since it coincided with a deep dip in the measured light, while for the rest of the day, the intensity stayed relatively stable. So the ODROID "saw" the eclipse much better than what was possible with the human eye, and made sure that I had some record of this awesome natural phenomenon. The following sections detail the hardware and software that I used to capture the memorable event.

Hardware and Software

ODROID-CI with Tinkering kit
Standard Linux OS for ODROID-CI
GPIO from the WiringPi package: http://bit.ly/IEq3UpF
Munin and nginx from the default repository

Connecting the parts

I used the breadboard that comes with the C1 Tinkering kit in order to build the circuit as shown in Figure 3. With this circuit, the voltage at AIN0 changes dependent on the resistance of the photo resistor. As a result, the voltage at AIN0 is coupled to the light intensity reaching the photo resistor.



Circuit schematic to connect the photo resistor to AIN0 of the ODROID

The CPIO program can then be used to read the analog input value and get a value between 0 and 1023 representing the voltage at the AIN0 port. The source code for the GPIO tool can be downloaded from Hardkernel's Github repository at http://bit.ly/1Eq3UpF. Typing the following lines into a Terminal window will download and compile the WiringPi software:

```
$ git clone https://github.com/hardkernel/wiringPi.
git
$ cd wiringPi
$ sudo ./build
```

To get regular measurements from the light sensor and have an easy-to-use interface in order to follow them over time, I installed a web server called nginx along with a monitoring software called munin. The installation of these two programs is easy:

```
$ sudo apt-get install nginx munin munin-node
```

I then added a section to the nginx configuration by creating a separate file called /etc/nginx/sites-enabled/munin:

```
server {
    listen 8000;
    root /var/cache/munin/www;
    index index.html index.htm;

    location / {
        try_files $uri $uri/ =404;
    }
}
```

To tell munin how to get data from the analog input port AIN0, I saved the following script as /etc/munin/plugins/ain:

```
#!/bin/sh

case $1 in
config)
cat <<'EOM'
graph_title Analog Input
graph_vlabel value
ain0.label AIN.0
graph_args -l 0 --base 1000
graph_scale no
graph_category GPIO
EOM
exit 0;;
esac

printf "ain0.value "
/usr/local/bin/gpio aread 0
```

All other configuration files were left unchanged. To start the required services and reload the configuration, type the following commands:

```
$ sudo service nginx stop
$ sudo service nginx start
$ sudo service munin-node stop
$ sudo service munin stop
$ sudo service munin-node start
$ sudo service munin start
```

You will then be able to access the munin reports at http://[ip-of-your-odroid]:8000, and after a couple of minutes, you should see the data from the analog input flowing into the graph.

Conclusion

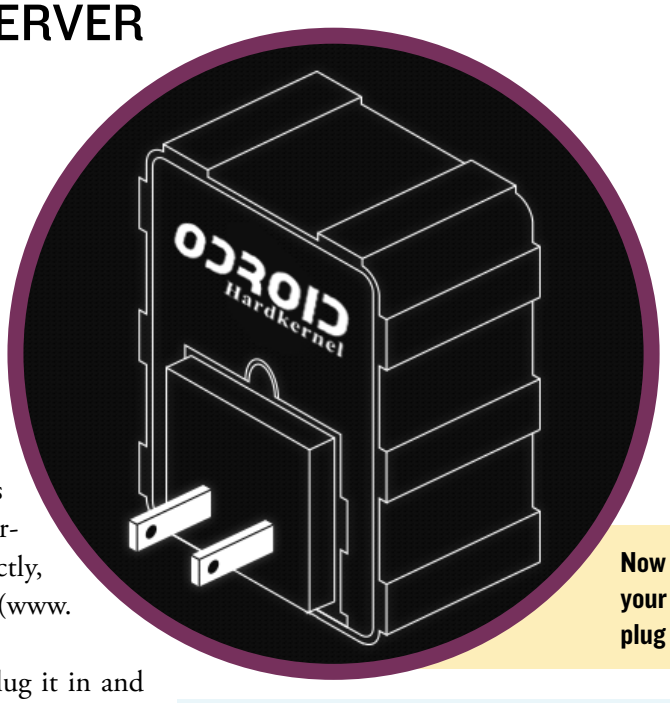
It is very easy to turn your ODROID-C1 into an autonomous observation unit. Because of the built-in analog input ports, it's just a matter of a few lines of code and very minimal tinkering on the breadboard in order to follow the changes of an environmental variable over time. Along with the light intensity, you could easily follow other values like temperature, air humidity or soil moisture of your potted plants. It's just a matter of which sensors you connect. I am planning to experiment a little with the sensors on the Hardkernel weather board, and I already ordered a soil moisture sensor. But those are just my ideas, and you can easily develop your own project using a similar method.

For questions and comments, please visit the original post at `http://bit.ly/1bfmbLZ`.

CONVERT A USB CHARGER INTO A TINY LINUX BOX

THE ULTIMATE TRAVEL SERVER

by Chris Robinson



Now you can disguise your ODROID as a wall plug and use it anywhere!

This tutorial will show you how to make a Linux computer that fits inside a modified USB charger plug, providing constant power in a very small package. It uses the Raspberry Pi-compatible ODROID-W development board and runs the Raspbian OS. The plug itself can be used internationally and has US, UK and EU attachments. Although Hardkernel no longer sells the ODROID-W board directly, they can still be purchased from Ameridroid (www.ameridroid.com).

The cool thing is that all you need to do is plug it in and that's it. You just leave it anywhere within wifi range, and it will run with constant power, and is small enough to be inconspicuous. There are quite a few applications for something like this, including: a personal file/media server, a silent motion alarm (when combined with the RPi Camera), a TOR relay, a pen-testing tool, a Bitcoin node, a personal VPN or just a general-use remote Linux system. This example uses a wifi adapter for connectivity, but alternatively you could use the innards of a USB ethernet adapter if you wanted a physical connection.

Parts and Materials

The total cost for the device was approximately $50 USD:

- ODROID-W single board computer (700Mhz, 512MB RAM)
- USB charger plug model "MD-ADP-0516UN001" which has UK/US/EU attachments
- USB Wifi adapter, which must be Raspberry Pi-compatible
- Insulating tape
- Scrap plastic for insulation
- Solder
- Some thin wire
- MicroSD card, which needs to be big enough to hold the operating system and whatever applications you intend to run

Tools

- Soldering iron
- Wire clippers/strippers
- Small hacksaw blade
- Fine file (optional)
- Helping hands stand (optional)
- Super glue

Software

Download the custom Raspbian OS image from the ODROID website at `http://bit.ly/1DKVp6b`. I tried the standard Raspbian image as well, which seemed to work, but I think that the ODROID-specific image is tailored to the hardware. Follow any one of the many guides on the Internet for how to write the .img file to your microSD card, or refer to the article on page 8 of the January 2014 issue of ODROID Magazine.

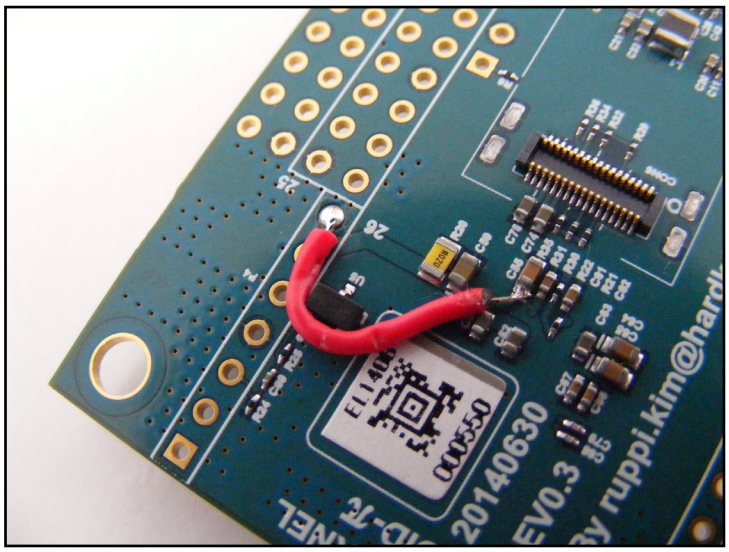
Follow the guide at `http://bit.ly/1PZ3xaI` in order to set up wifi. Since there is no USB port on the board by default, you will have to solder one before connecting the wifi adapter. This is only required in order to set everything up, and may be removed afterwards.

Hardware

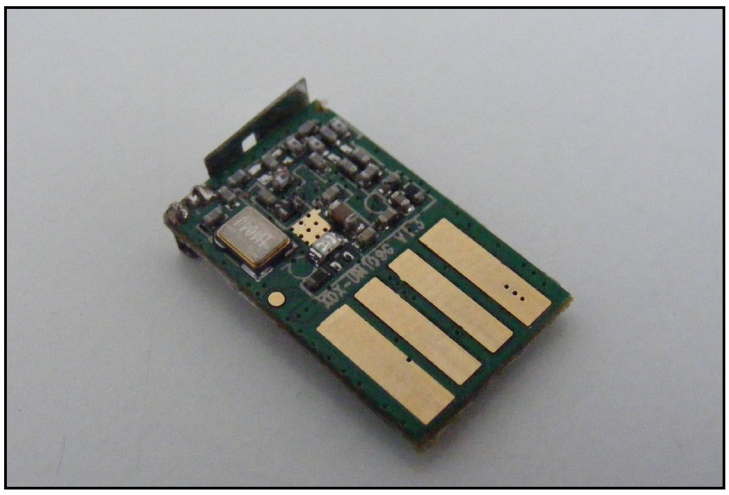
Disassembling USB chargers is potentially dangerous and could harm or kill you if you don't take necessary precautions. Be careful when dealing with live electricity, and make sure to unplug the charger before performing the next set of steps.

Place a little jumper wire between the adapter input pin

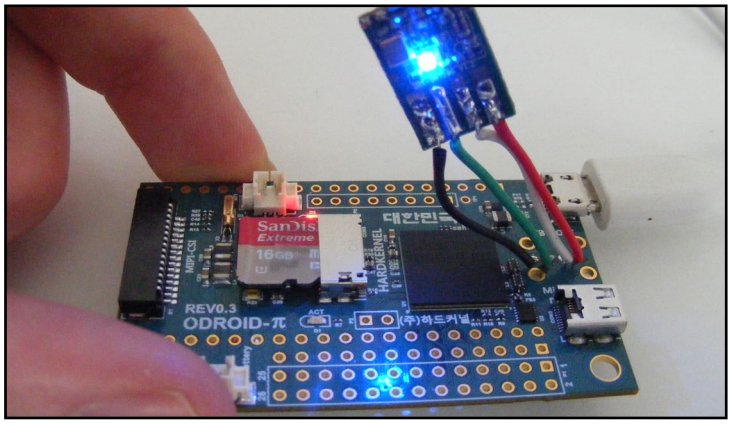
and the micro-USB power input, as shown in Figure 1. The ODROID-W has some power stability issues that are solved with this fix. More info about this modification may be found on the ODROID forums at http://bit.ly/1ERSbng.



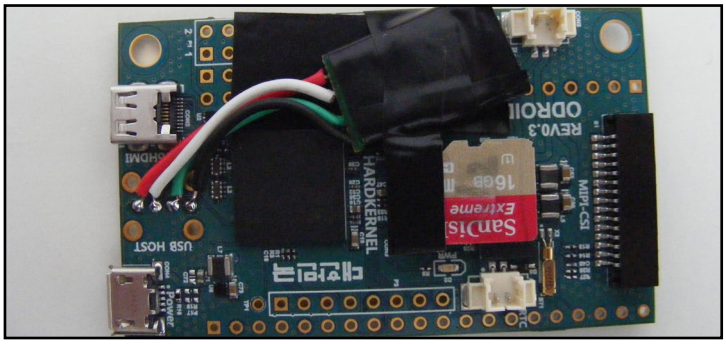
Remove the plastic and metal casing from around the wifi adapter. You should be left with something that looks like Figure 2.



Cut, strip and tin some wires and solder them between the board and the contacts on the wifi card, as shown in Figure 3. Power it up to verify that it works properly before moving on to the next step.

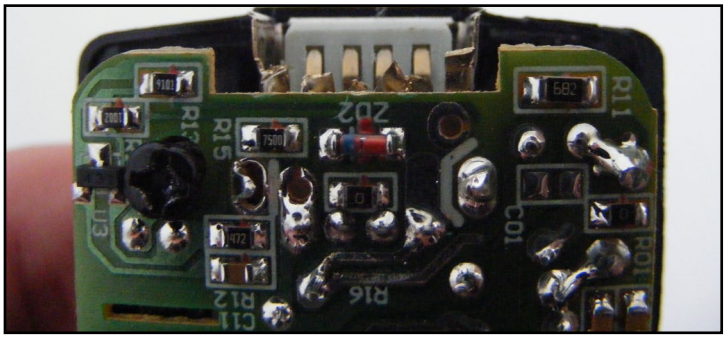


Insulate the connection using insulating tape, then fold it over so that it fits in the empty space, as shown in Figure 4.

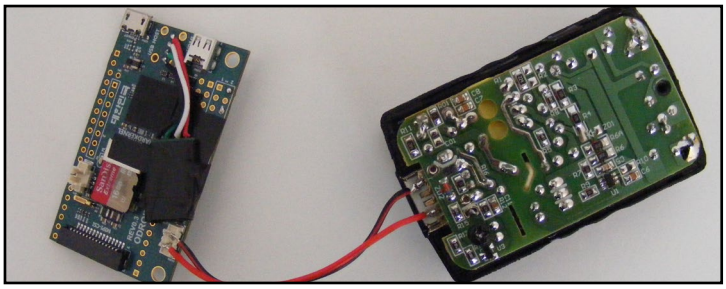


Take apart the USB charger by using a small hacksaw blade to cut around all four edges, then pry it in half. Going slow here helps not to accidentally saw into the components inside, which necessitates starting over with another charger. You will be left with something that looks like Figure 5.

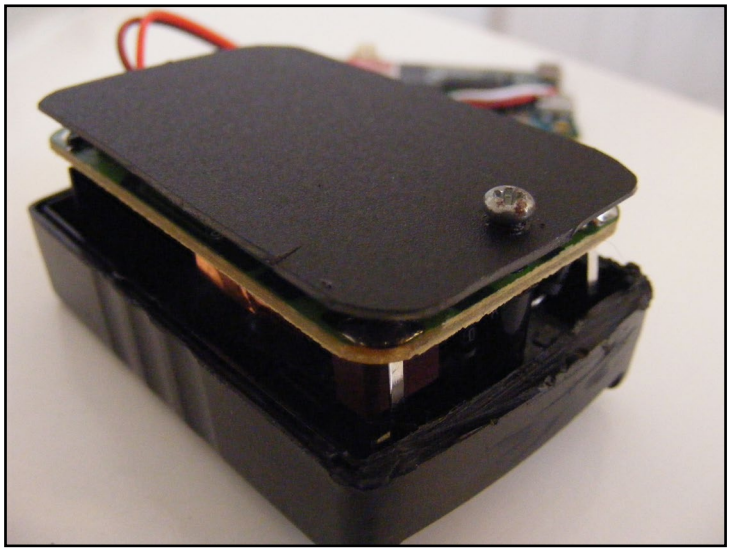
Instead of modifying the components on the board, I decided to cut away the front of the USB port, revealing the contacts, as shown in Figure 6.



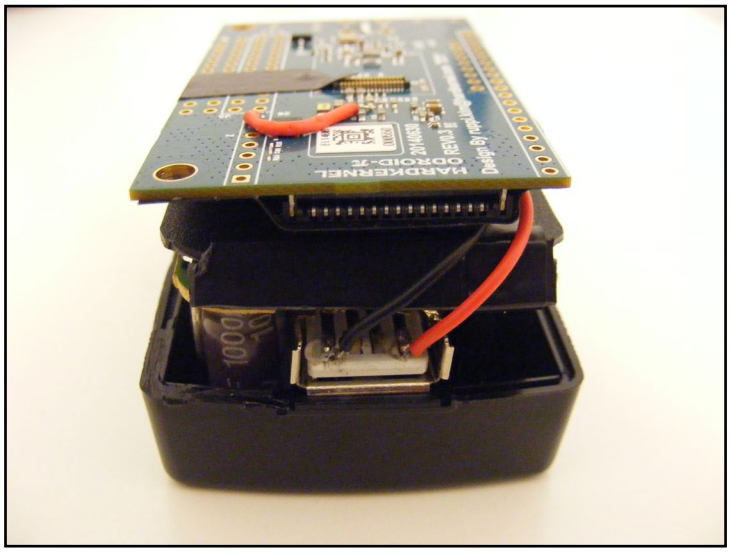
Cut, strip and tin a red and black wire, and solder them to the USB charger. The other half may be soldered directly onto the ODROID's battery connector. This helps save space instead of using the Micro USB port, as shown in Figure 7.



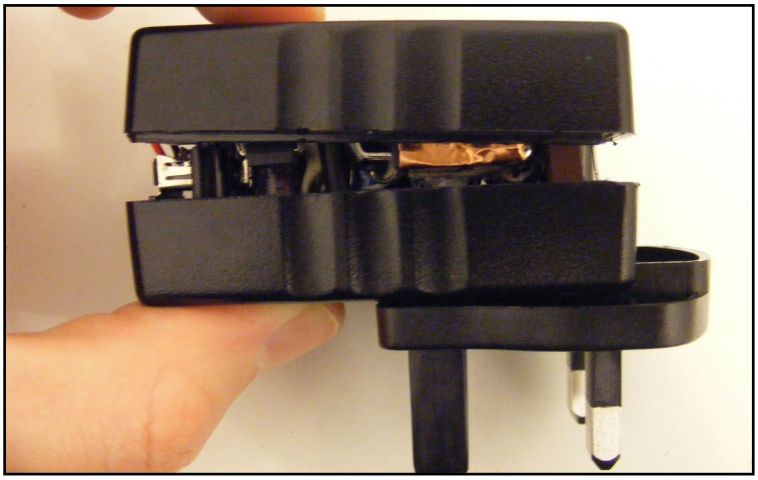
Cut some scrap plastic and screw it onto the exposed USB charger in order to insulate the electricity from the charger. I used 2mm thick plastic which seemed to work well, as shown in Figure 8.



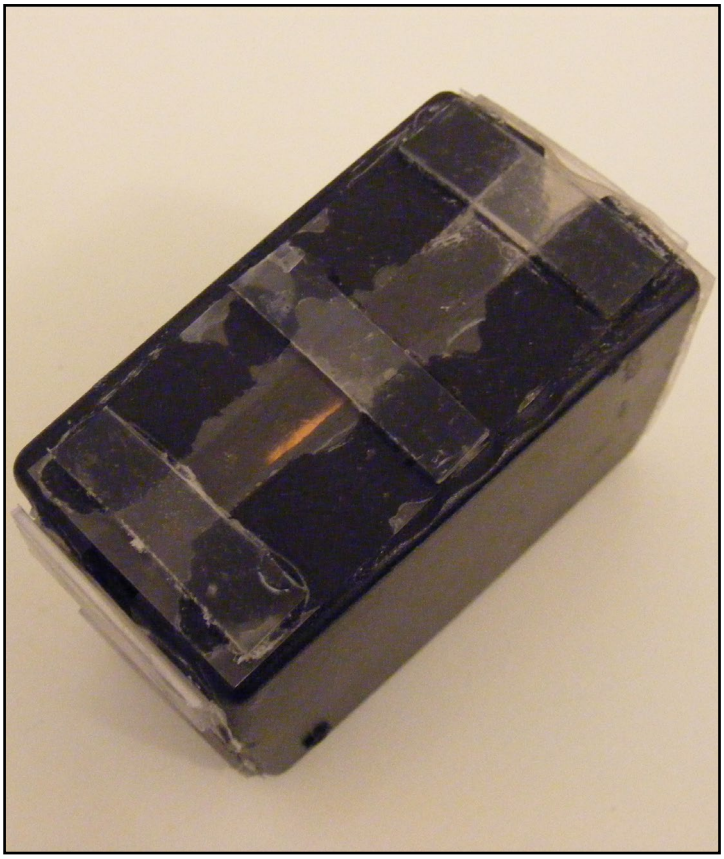
Stack everything up so that it will fit inside the case.



You'll notice that there is a gap between the two halves of the case, as seen in Figure 10, so it will be necessary to cover that space.



I used thin plastic to cover the exterior, and then thicker plastic to provide support, as seen in Figure 11. It ended up being fairly strong.



Finish the exterior with vinyl sheeting, shrink-wrap plastic, or another material to make it less conspicuous. You now have a little Linux plug-sized computer!

If you have questions or comments, please refer to the original article at http://bit.ly/1BxDDkk.

MEET AN ODROIDIAN

DANIEL MEHRWALD (@AREASCOUT)
RETRO EMULATION AND GAMING AFICIONADO

edited by Rob Roy

From programming to field service, you can count on Daniel to tackle all sorts of problems!

Please tell us a little about yourself.

My name is Daniel Mehrwald. I am from Austria and 42 years old. I am a highly technically-oriented guy, with an interest in everything to do with electronic devices, computers, technology in general, electric cars, and state-of-the-art inventions. I am very interested in these domains and "slurp" them up if there is anything new to it. On the other hand, I love nature, hiking, bicycle, diving and swimming, if time and money allow it. For my job, I learned how to be an electrician by doing programmer logic controller (PLC) programming, which eventually led me to field service, working for very specialized big companies doing repairs and maintenance tasks.

Daniel diving in the Egyptian Red Sea and enjoying the amazingly clear water

How did you get started with computers?

It was in the mid 1980s when I got my first computer, which was the great Commodore 64. At that time, the cold war between East and West was still ongoing, and revolutionary home computers were totally new on the market. Before that, the TV was a static non-interactive medium, but with a home computer, you could control the picture on the TV, which was really amazing. I wrote my first BASIC program, by typing in small games from computer magazines.

Later, I wrote my own programs, such as graphic demos using Koala Paint, and composing music with SoundMonitor. I exchanged my programs with other Commodore 64 owners via cassette tape and floppy disc. That was real power, and was a hobby shared by millions of people. Many of them built their own demos or application intros in order to send messages to people internationally. The messages were short but meaningful, with some even starting with the words "Hello boys and girls all over the world". Wow, it was mind-blowing, and gave me the feeling that all humans on earth are united. What a great idea!

To fall behind such a revolution was not an option for me. Typical messages protested against the war and the nuclear threat and human stupidity. One of my favorites was the famous Trap demo (http://bit.ly/1GA2c64). Remember, lots of those messages were never heard before on local TV stations at this time. It was a totally new way to learn and communicate, and what people did with their computers was mostly art on a large scale. The ongoing development in theses technologies are still influenced from that time. I eventually bought an Amiga 500, with which I spent many hours. The other computers that I bought were mostly Intel-based PCs. I also owned an Apple computer, but my first Commodore 64 will always have a special place in my heart.

What drew you to the ODROID platform?

My first single board computer was the BeagleBoard. I also experimented with Gumstix, and had two Raspberry Pis. The ODROID was a logical step forward, since the processing power and GPU capability was great. It took some time on the X2 and U2/U3 until the GPU drivers were in a useful state, but I helped the progress with the inclusion of the LIMA driver in the Versatile Commodore Emulator (VICE) and RetroArch.

Which ODROID is your favorite?

I really like the U3 because it has a good price/performance ratio, but the X2 is my favorite so far. It has very good hardware, the CPU temperature was very low due to the large heatsink, and it has a lot of USB ports. I also own an XU3, so maybe this will become my favorite in the future.

Describe your ODROID setup and how you use it.

My ODROID is connected directly to my TV in the living room. I use SSH and the serial debug interface to connect from my PC to the ODROID. For programming, I use a remote desktop connection, and run Eclipse IDE from it because it has a nice import for Makefiles. From there, I can run and debug applications comfortably and watch the output on my TV.

You are very generous on the ODROID forums with sharing your knowledge of gaming and Linux programming. How did you become so proficient?

The days of information hoarding for only a few "powerful" people at the top is outdated. Sharing knowledge is the future! If you spend a certain amount of in-depth time with computers and programming, then you can become proficient. For me, it's just the years and time spent playing around with them. What maybe makes me unique is that I didn't learn programming or how to use computers in school or on a job. I haven't had any special training for it. 90% of my knowledge, I learned by myself. I just tried everything out, which was empirical research, plus trying to think logically. That was really the hard way. Things that people learn within minutes in university, I had to find out through experimentation, and sometimes it took me a week or months. However, the Internet and friends helped me out a lot as well.

Which projects are you involved in for the ODROID?

I have worked on Doom3, RetroArch, Nintendo 64, and MAME. All of them need service from time to time.

What hobbies and interests do you have apart from computers?

As I mentioned previously, I like hiking, cycling, diving, and swimming. I also enjoy reading books, using Photoshop, and watching science fiction movies. This year, I'm looking forward to the new Star Wars movie coming out. I spend time with all of my hobbies occasionally, depending on what I feel like doing.

What type of hardware innovations would you like to see for future Hardkernel boards?

In the past, I have had a lot of ideas, but now I am perfectly happy. ODROIDs have met all of my needs.

What advice do you have for someone want to learn more about programming?

First, do it for your own enjoyment, and not for someone else. It should be fun, and it's just about continuing to try things until something works. Some people give up very quickly, but it's better to stay with it. It's a great feeling when, after 2 or 3 days of research, you achieve your goal. Of course, it's best to start with a high-level programming language.

Do you have anything else to share with our readers?

There is an awakening, have you felt it?

Daniel is certainly a masterful Star Wars Photoshop Jedi!