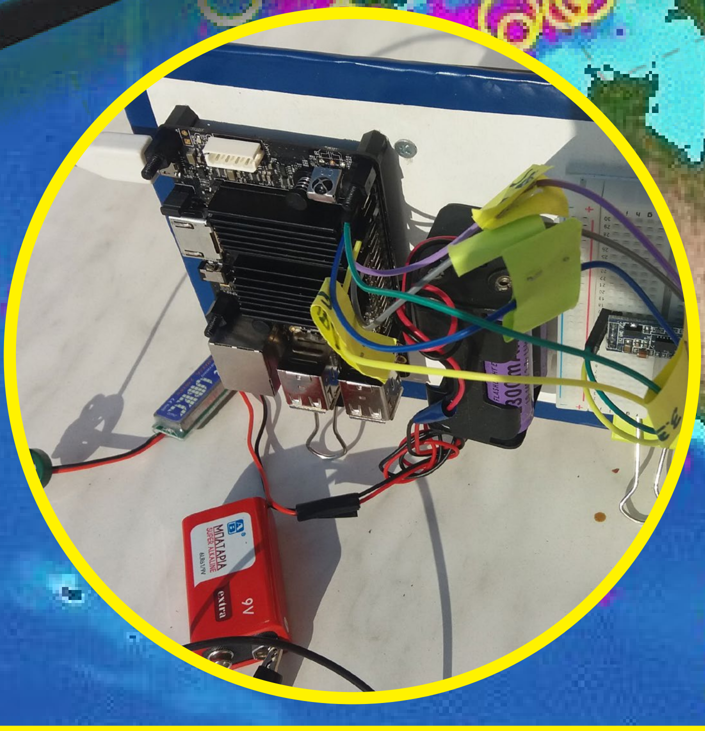
Deep Diving Inside ADB • Geiger Counter • Lakka • Gogs • I/O Extenders

ODROID Magazine

Year Four
Issue #43
Jul 2017

Sensor tech: ODROID Seismograph



Measuring Seismic Acceleration using the ODROID-C2

A Smart house with your ODROID by your side

Use your own ODROID-XU4 As A Map Server

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-C2 and ODROID-XU4 devices 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

Suche – Pollin Electronic
www.pollin.de/shop/suchergebnis.html?S_TEXT=odroid&log=internal
Elektronik, Technik und Sonderposten zu Superpreisen!
mein Konto
Kontakt & Service
Versand & Bezahlen
Katalog
Newsletter
Gutscheine
Merkliste | Neu registrieren | Anmelden
Direktbestellung
Einkaufswagen
Suchen nach:
Suchbegriff oder Bestellnummer
Los
Kategorie wählen
Filtereinstellungen
[-] alle Filter entfernen
Kategorie
[+] Bausätze / Module (17)
[+] Stromversorgung (1)
Preis
Ihre Vorteile
Innovative Produkte
Optimales Preis-/Leistungsverhältnis
Kompetente Beratung
Schnelle Lieferung
Noch Fragen?
Wir sind gerne auch persönlich für Sie da!
+49 (0) 8403 / 920-920
Sicher Einkaufen

Sie sind hier: Startseite > Suche

Suchergebnis Ihre Suche nach "odroid" ergab 18 Treffer

sortieren nach: Relevanz (gewählt)

1 - 18 von 18 Ergebnissen

ODROID-U3 BACKUP BATTERY
RTC Lithium-Backupbatterie für den ODROID XU, XU+E, XU Lite und U3. Wird direkt auf den RTC-Port des ODROID gesteckt.
2,95 €*
Best.Nr. 810 319

ODROID-U3 Einplatinen-Computer, Cortex-A9 QuadCore, 2 GB
Kleiner als eine Scheckkarte aber riesig in der Leistung! Der ODROID-U3 verfügt über eine leistungsstarke SAMSUNG QuadCore-CPU, 2 GB Arbeitsspeicher und
69,95 €*
Best.Nr. 810 264

ODROID-U3 ODROID-UPS, USV, LiIon, 3 Ah
Unterbrechungsfreie Stromversorgung mit zwei LiIon-Akkus. Das Board besitzt die gleichen Außenmaße wie der ODROID-U3 Einplatinencomputer und kann direkt per
31,95 €*
Best.Nr. 810 328

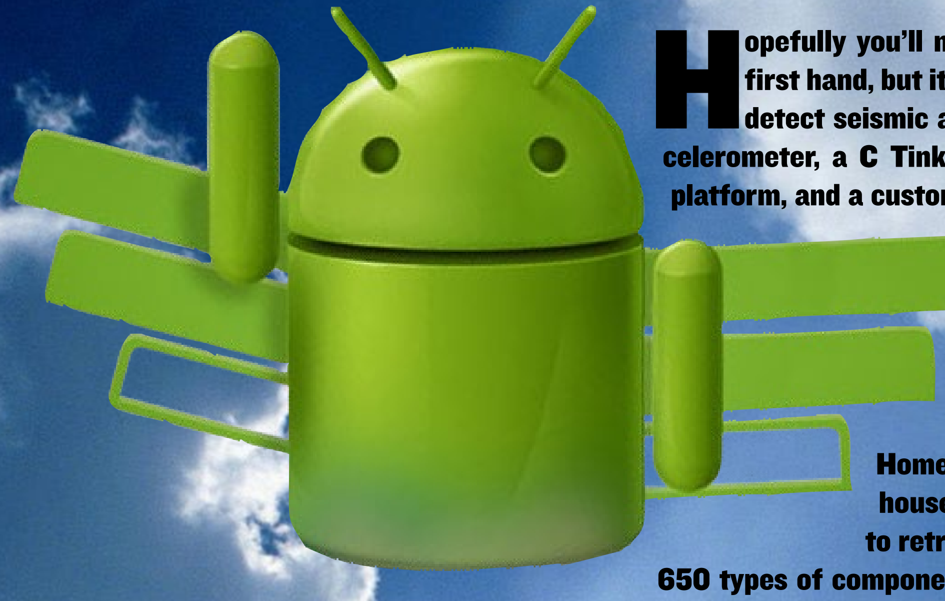
ODROID-U3 eMMC Modul, 64 GB, mit Linux
Superschnelles NAND-Speichermodul für den ODROID-U3 Einplatinencomputer. Das Linux-Betriebssystem XUBUNTU ist fertig aufgespielt und kann direkt

ODROID-U3 USB-UART MODULE, Schnittstellenkarte
USB/UART System-Interface (IC: CP für die ODROID Einplatinencomput X2, U2, U3, XU, XU+E, XU Lite). I Entwicklung und Debugging.
15,95 €*
Best.Nr. 810 326

ODROID-U3 CASE Kunstst
Passgenaues Gehäuse mit Lüftungsschlitzen für den Maße (LxBxH): 87x52x29
4,95 €*
Best.Nr. 810 317

ODROID-U3 eMMC Android
Superschnelles N den ODROID-U3 Betriebssystem aufgespielt un
39,95 €*
Best.Nr. 810

ODROID-U Linux
Supersch den OD Linux- berei



Hopefully you'll never have to experience an earthquake first hand, but it's nice to know that an ODROID can help detect seismic activity in your area! Using a simple accelerometer, a C Tinkering Kit, WiringPi library, ThingSpeak platform, and a custom Python script, a C2 can be turned into a miniature geographic laboratory that outputs the magnitude of waves that occur nearby. Our regular contributor Miltiadis details this amazing project so that you can build one of your own.

Home automation is virtually standard in new houses these days, and Adrian shows us how to retrofit an existing house to connect to over 650 types of components in order to have completely centralized control over smart devices from any ODROID device. Coupled with one of Hardkernel's touchscreens, it provides an inexpensive alternative to installing a commercial automation package. Nanik continues his series on the Android Debug Bridge (ADB), José presents a guide to using an ODROID-XU4 as a map server, @jojo details his homemade Geiger counter, @redrocket gives an overview of Gogs, a GitHub/GitLab alternative, and we learn about Hardkernel's new header extenders.

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/1ypImXs.
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

ameriDroid.com
High-Performance Embedded Computers

Hundreds of products available online for the professional developer and hobbyist alike

ODROID-XU4

ODROID-C1+

ODROID-C0

OWEN ROBOT KIT

ODROID-C2

VU7 TABLET KIT

Hardkernel's EXCLUSIVE North American Distributor

Rob Roy, Chief Editor

I'm a computer programmer 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`.

Bruno Doiche, Senior Art Editor

Kept playing Exagear games, but is tempted to also trade his playstation 3 for a playstation 4 to enjoy his so awaited release of the game he was waiting for it. Elite Dangerous! The dog is enjoying the best vacation at his house while his father in law recovers from a broken arm.

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.

Nicole Scott, Art Editor

Nicole is a Digital Strategist and Transmedia Producer specializing in online optimization and inbound marketing strategies, social media management, and media production for print, web, video, and film. Managing multiple accounts with agencies and filmmakers, from web design and programming, Analytics and Adwords, to video editing and DVD authoring, Nicole helps clients with the all aspects of online visibility. Nicole owns anODROID-U2, a number of ODROID-U3's, and Xu4's, and looks forward to using the latest technologies for both personal and business endeavors. Nicole's web site can be found at http://www.nicolecscott.com.

James LeFevour, Art Editor

I'm a Digital Media Specialist who is also enjoying freelance work in social network marketing and website administration. The more I learn about ODROID capabilities, the more excited I am to try new things I'm learning about. Being a transplant to San Diego from the Midwest, I am still quite enamored with many aspects that I think most West Coast people take for granted. I live with my lovely wife and our adorable pet rabbit; the latter keeps my books and computer equipment in constant peril, the former consoles me when said peril manifests.

Andrew Ruggeri, Assistant Editor

I am a Biomedical Systems engineer located in New England currently working in the Aerospace industry. An 8-bit 68HC11 microcontroller and assembly code are what got me interested in embedded systems. Nowadays, most projects I do are in C and C++, or high-level languages such as C# and Java. For many projects, I use ODROID boards, but I still try to use 8bit controllers whenever I can (I'm an ATMEL fan). Apart from electronics, I'm an analog analogue photography and film development geek who enjoys trying to speak foreign languages.

Venkat Bommakanti, Assistant Editor

I'm a computer enthusiast from the San Francisco Bay Area in California. I try to incorporate many of my interests into single board computer projects, such as hardware tinkering, metal and woodworking, reusing salvaged materials, software development, and creating audiophile music recordings. I enjoy learning something new all the time, and try to share my joy and enthusiasm with the community.

INDEX

HOME AUTOMATION WITH HOME ASSISTANT

A SMART HOUSE WITH YOUR ODROID BY YOUR SIDE

by Adrian Popa

There comes a time in everyone's life when you want to put some things in order and have simple access to complex solutions. For example, maybe you have several scripts taking care of various problems (like turning a heater on/off, taking pictures with your security cameras, handling presence detection, etc), but you're the only one who can manage them because they require maintenance through SSH, or through some old-looking web page. I too have reached the same place in my life, and have to look for an "umbrella" solution to manage all my personal automations and offer easy access for my family.

I was thinking of building a web dashboard to fit my needs, but I hate web development. I'm somewhat lazy and my sites are not good looking at all. Furthermore, it needed to be functional on all sorts of devices and screen sizes, and also future-proof. Fortunately, I spent enough time looking around until I found the perfect solution - Home Assistant (`http://bit.ly/2hlOPOE`) - HA for short.

Home Assistant is an open-source home automation platform built on Python 3 that supports over 650 components, which are modules that facilitate interaction with things like physical "smart" switches, relays, lights, sensors, network devices (TVs, routers, and cameras), software (like Kodi, MPD, and Transmission), network services (like weather), but also allows you to add your own custom components. All of the major home automation brands and technologies, like Hue, Nest, IKEA, Vera, ZigBee, and MQTT are present, and a complete list of components can be found at `http://bit.ly/2sWJsPy`.

Apart from the components, the platform has a dashboard-like web interface and an automation engine where you can combine data from different components and generate an event. For example, if it's Monday-Friday between 8:00 - 15:00 and the outside weather is sunny, and the outside temperature is above 30C, and there is no chance of rain, and the outside sprinklers have been off for at least 4 hours, then turn on the sprinklers for 20 minutes. The only complicated thing in the automation above is having a way to turn your sprinklers on and off - the rest is provided by existing components and Home Assistant's automation engine. Other use cases might include locking and unlocking the front door when a specific person connects to the wifi (although I wouldn't do this personally), or starting the air conditioning automatically when the system detects you're coming home from work. There are more use cases in the 1-hour video at `http://bit.ly/2t0GgCI`. If you're familiar with Tasker for Android or IFTTT, then Home Assistant is the equivalent for your home.

Now your house will be smarter than your friends!

Installation

You can install Home Assistant on any ODROID device. Depending on how many automations you plan to have, you could use a C1 for a light setup, or even an XU4 for large homes and complex rules which might involve face recognition. I'm using it on a C2 which doubles as a Kodi player without issues.

We're going to do the "virtualenv" installation, which means that all the required python modules will be installed in a specific directory and will not interfere with system modules. We will also use a distinct user for Home Assistant. There are also Docker images available. The complete instructions with comments are available at `http://bit.ly/2t0iaYC`.

```
$ sudo apt-get update
$ sudo apt-get dist-upgrade
$ sudo apt-get install python-pip python3-dev
$ sudo pip install --upgrade virtualenv
$ sudo adduser --system homeassistant
$ sudo addgroup homeassistant
```

```
  $ sudo usermod -G dialout -a homeassistant
  $ sudo mkdir /srv/homeassistant
  $ sudo chown homeassistant:homeassistant /srv/home-
assistant
  $ sudo su -s /bin/bash homeassistant
  $ virtualenv -p python3 /srv/homeassistant
  $ source /srv/homeassistant/bin/activate
  (homeassistant)$ pip3 install --upgrade homeassis-
tant
  $ exit
```

In order to start and manage the process, it's best to create a systemd service to handle it:

```
  $ sudo vi /etc/systemd/system/homeassistant.ser-
vice
  [Unit]
  Description=Home Assistant
  After=network.target time-sync.target
  Requires=time-sync.target

  [Service]
  Type=simple
  User=%i
  ExecStart=/srv/homeassistant/bin/hass -c "/home/ho-
meassistant/.homeassistant"

  [Install]
  WantedBy=multi-user.target
```

In order to start Home Assistant, simply start its service:

```
  $ sudo service homeassistant start
  $ sudo service homeassistant enable
```

Note that if you will be using components that need HTTPS, you will need to have time correctly set up at boot, so that the certificates are valid. The service startup depends on systemd-timesyncd, which in turn depends on ntp *not* being installed:

```
  $ sudo apt-get remove ntp
  $ sudo service systemd-timesyncd restart
  $ sudo systemctl enable systemd-timesyncd
```

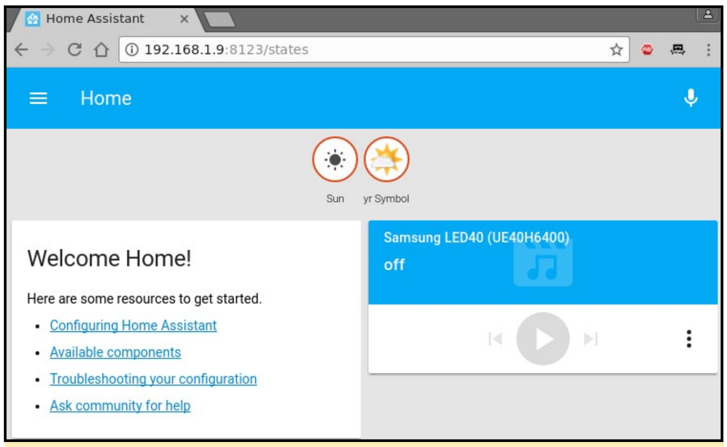
In case of problems, you will be able to review the logs through journalctl:

```
  $ sudo journalctl -u homeassistant -f
```

Once the process starts, you will be able to connect to `http://odroid-ip:8123/`. Note that the first startup (or a startup following an update) might be slower, so leave it run for a few minutes until accessing the web interface. Home assistant also provides a native app for IOS (`http://apple.co/2tYi2WI`), while for Android clients you can pin the page as a homescreen launcher (Chrome -> navigate to http://odroid-ip:8123 -> Menu -> Add to homescreen).

The configuration file

In order to set up components and configure your installation, you'll have to work a lot with Home Assistant's configuration file(s). Hopefully, in a future version you might be able to handle the configuration directly from the web interface, but for now, you'll need a text editor. The main file is /home/homeassistant/.homeassistant/configuration.yaml. Its format is YAML - which stands for "Yet Another Markup Language". Like Python, it uses white space (not tabs!) to delimit sections of code. By default it uses a two space indentation for nested sections. In case you get into trouble, you will receive error messages when starting the service. You can validate the syntax with a service like `http://www.yamllint.com/` which will let you know where you went wrong. There is also a troubleshooting guide at `http://bit.ly/2tDHMsa`.



Welcome Home!

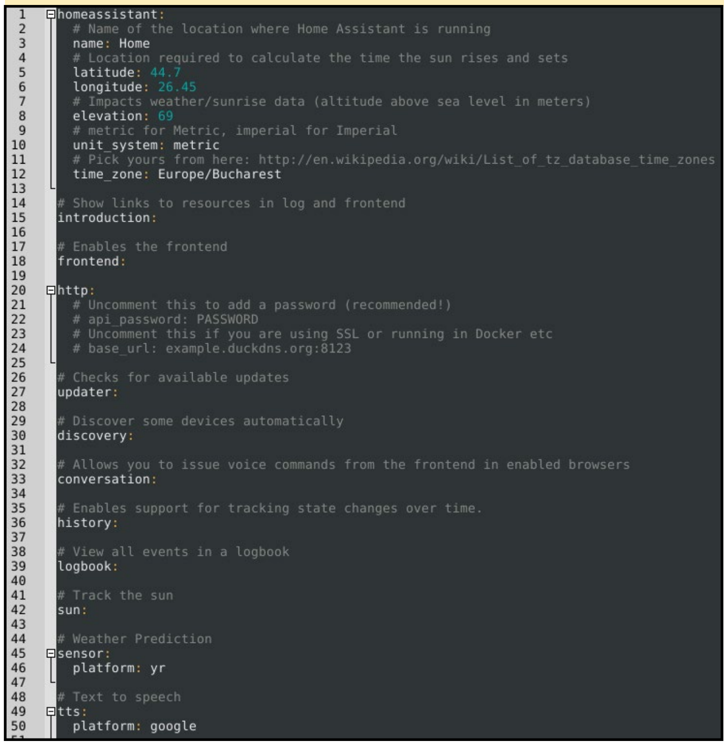
Here are some resources to get started.

- Configuring Home Assistant
- Available components
- Troubleshooting your configuration
- Ask community for help

Figure 1 - Home Assistant startup page

Once you've made changes to the configuration file, you will need to restart the homeassistant service to apply those changes. You can do this either from the shell with sudo service homeassistant restart, or from HA's web interface, by clicking the top left icon, selecting the "Configuration" icon and calling the "Re-

Figure 2 - The default configuration



```
homeassistant:
  # Name of the location where Home Assistant is running
  name: Home
  # Location required to calculate the time the sun rises and sets
  latitude: 44.7
  longitude: 26.45
  # Impacts weather/sunrise data (altitude above sea level in meters)
  elevation: 69
  # metric for Metric, imperial for Imperial
  unit_system: metric
  # Pick yours from here: http://en.wikipedia.org/wiki/List_of_tz_database_time_zones
  time_zone: Europe/Bucharest

# Show links to resources in log and frontend
introduction:

# Enables the frontend
frontend:

http:
  # Uncomment this to add a password (recommended!)
  # api_password: PASSWORD
  # Uncomment this if you are using SSL or running in Docker etc
  # base_url: example.duckdns.org:8123

# Checks for available updates
updater:

# Discover some devices automatically
discovery:

# Allows you to issue voice commands from the frontend in enabled browsers
conversation:

# Enables support for tracking state changes over time.
history:

# View all events in a logbook
logbook:

# Track the sun
sun:

# Weather Prediction
sensor:
  platform: yr

# Text to speech
tts:
  platform: google
```

start" option from the "Server Management" section. The video at `http://bit.ly/2sAmD3F` shows some tips you should consider when editing the configuration.

If you plan on using HA from outside the LAN (e.g. from the Internet), you have several options. One of them is to enable HTTPS support and forward port 8123 on your router. This gives you encryption, but exposes your installation to the internet (and there might be vulnerabilities that could allow attackers take control of your system/LAN). A second option (which I prefer) is to set up a VPN on your router (or even on your ODROID) that allows you to connect and access HA (and other LAN resources) securely.

If you want to use HTTPS, in order for all features to work you will need to supply valid SSL certificates (not self-signed). In order to get valid certificates you will need to have a public DNS name (e.g. by using a dynamic DNS service like duckdns.org) and use letsencrypt.org to set up a valid SSL certificate for your installation. Step by step details can be found in the video at `http://bit.ly/2tY6LGb`. If you must use self-signed certificates, there is a guide available at `http://bit.ly/2t0ObzH`.

Regardless of access mechanism (http or https), you will want to set up a password. HA doesn't support multiple user accounts, but you can set an API Password that you will need to log into the web interface. The best way to do this is to create a file that will keep all your sensitive data (like passwords and URLs), name it "secrets.yaml" and reference it in the configuration.yaml file.

```
 $ cat /home/homeassistant/.homeassistant/secrets.
yaml
 api_password: odroid
 $ cat /home/homeassistant/.homeassistant/configura-
tion.yaml
 …
 http:
   api_password: !secret api_password
 …
```

Now, when you will restart HA, you will be asked for a password. More details about secrets may be found at `http://bit.ly/2rLGEkV`.

In order to get acquainted with how HA configuration works, we will set up some components. I want to set up weather, some IP cameras, Kodi and MPD, presence detection based on WiFi and also a 1-wire temperature sensor connected to the ODROID.

Weather from Darksky

There are several weather providers already integrated in HA (`http://bit.ly/2t4l1Rh`), so you can pick your favourite. I'm going with DarkSky (`http://bit.ly/2t4gq0S`), which provides quite accurate forecasts for my area. You should consult the component's help page for details about configuration and which variables you can use. You will need to register with Dark Sky and get an API Key which will let you make 1000 calls per day

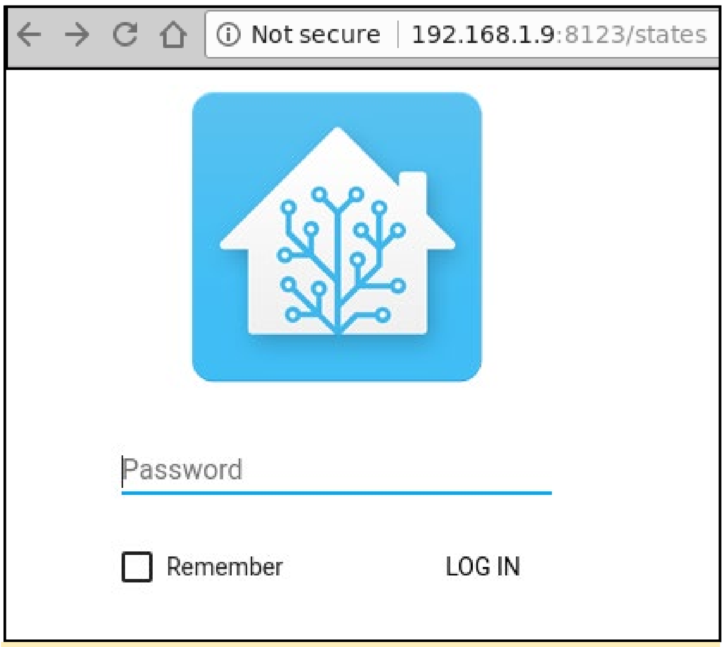


Figure 3 - Authentication

for free. It's best to save this API Key inside your secrets.yaml file (replace with your own key):

```
darksky_api_key: 87f15cbb811204412cc75109777ea5cf
```

The configuration has several variables, most of which are optional, however, under configuration.yaml, under the sensor section you would have the following (feel free to delete the "platform: yr" entry):

```
sensor:
  - platform: darksky
    api_key: !secret darksky_api_key
    name: Dark Sky
    monitored_conditions:
      - summary
      - precip_type
      - precip_probability
      - temperature
      - humidity
      - precip_intensity
      - wind_speed
      - pressure
      - wind_bearing
      - apparent_temperature
      - icon
      - minutely_summary
      - hourly_summary
      - temperature_max
      - temperature_min
    units: si
    update_interval: '00:15'
```

The code is mostly self-explanatory. It configures a new platform of the type "darksky", with a specific name (optional) and api_key (required) and pulls a set of parameters (monitored_conditions) from the weather provider every 15 minutes. Your actual location is taken from the latitude/longitude parameters under homeassistant, so make sure that's correct. After you restart the homeassistant service, you should be able to see the monitored variables as badges on the top of your window. Clicking on a badge will show you how that particular value has changed over time.

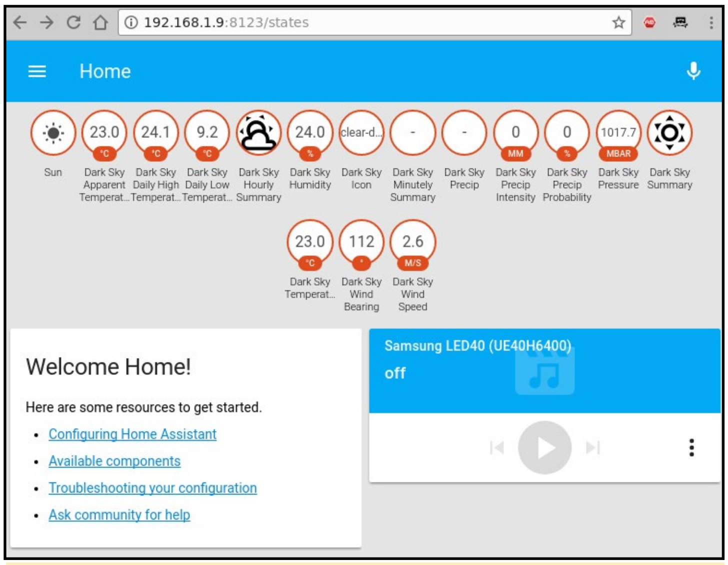


Figure 4 - Weather data

Viewing IP cameras

HA supports a lot of cameras (`http://bit.ly/2t4DtsD`), including reading data from a file, which could be used to display a graph, or visual data generated by other tools. We will be using the Generic MJPG Camera (`http://bit.ly/2t4tIKM`) component and the Local File (`http://bit.ly/2s4Y5w4`) component.

The camera we want to monitor is available at `http://bit.ly/2t4cHkc` (it's a public webcam), which we should add to the `secrets.yaml` file.

```
  camera1_stream_url: http://iris.not.iac.es/axis-
cgi/mjpg/video.cgi?resolution=320x240
  camera1_still_url: http://iris.not.iac.es/jpg/im-
age.jpg
```

The configuration part inside configuration.yaml looks like this for both cameras:

```
camera:
  - platform: mjpeg
    mjpeg_url: !secret camera1_stream_url
    still_image_url: !secret camera1_still_url
    name: Observatory in Spain
  - platform: local_file
    file_path: /tmp/tux.jpg
```

As usual, you will need to restart the HA service to reread the configuration (this might be a good time to comment out the "introduction" component as well). Note that when you click on a webcam you will see a live feed, otherwise the still image is updated every 10 seconds.

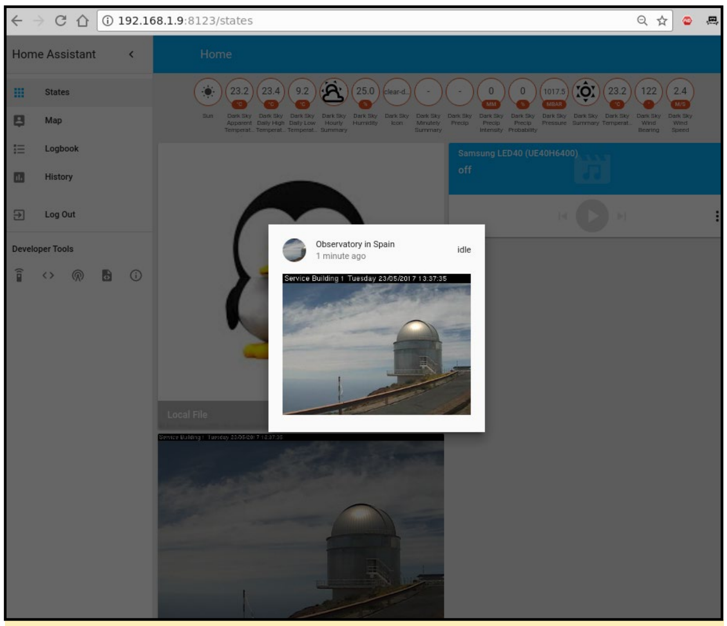


Figure 5 - Webcams!

So, what can you do with these configured webcams apart from looking at them? Well, you can use them with other components such as OpenCV (`http://bit.ly/2s4UUEJ`) to generate triggers when certain faces are seen, or Seven Segments Display (`http://bit.ly/2sAbOP0`), which can take readings of various digital displays.

Kodi and MPD

To configure media players, you can look under the Media Player component list at `http://bit.ly/2s0IAtQ`. To configure Kodi (`http://bit.ly/2sA5qr6`), you will need to enable the "Allow remote control via HTTP" option (`http://bit.ly/2t4cYne`) and set an appropriate username and password first. To do so, add the user and password to the secrets.yaml file:

```
kodi_user: kodi
kodi_pass: kodi
```

Then, edit configuration.yaml:

```
media_player:
  - platform: kodi
    host: 192.168.1.140
    name: Kodi Livingroom
    username: !secret kodi_user
    password: !secret kodi_pass
```

To configure MPD, assuming that you already have a MPD server in your network, add the MPD component (`http://bit.ly/2s5sbzE`) and add the password to secrets.yaml:

```
mpd_secret: mpd
```

And next, edit configuration.yaml:

```
media_player:
...
  - platform: mpd
    host: 192.168.1.140
    name: MPD Living
    password: !secret mpd_secret
```

After you restart Home Assistant, you will get the two new media players and be able to see their state (playing/stopped), control volume and even change the current playlist or use the text-to-speech component to have the media player "speak" what you want.

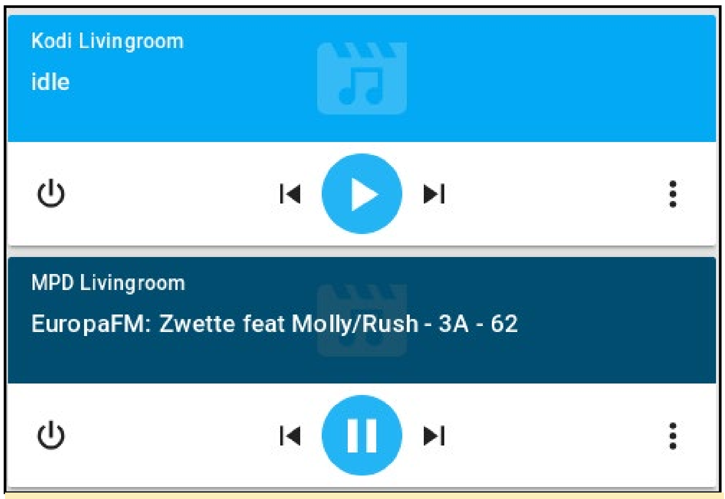


Figure 6 - Media players

Presence detection

The presence detection components (http://bit.ly/2t0Gt8H) try to track people's locations so that you can apply geofencing rules (e.g. do something if a person enters or leaves a location). Usually tracking is done by detecting devices connected to a router (via wifi), or via bluetooth proximity (http://bit.ly/2s0Sqfw), or by using location services such as Owntracks (http://bit.ly/2rLQdR1).

We will use a router-based tracker that, depending on your router, periodically connects to the management interface of your router, lists the ARP table, and discovers which devices are connected. A lot of router types are supported, from high-end vendors like Cisco, to consumer-grade routers like Asus, Netgear and TP-Link. Even open-source firmwares are supported, like OpenWRT, DD-WRT and Tomato.

We will be using an Asus router with SSH enabled, so we need the ASUSWRT component: http://bit.ly/2s4T32Q. You can chose to login with username/password or setup an SSH key and log in with a key instead. Note that certain firmware versions enable security measures which limit the number of SSH connections, and can blacklist your IP if a lot of connections are initiated.

As usual, we will set private data (such as the path to the key or the ssh password) in the secrets.yaml file:

```
router_user: admin
router_password: my_secret_password
```

Inside configuration.yaml add the following section:

```
device_tracker:
  - platform: asuswrt
    host: 192.168.1.1
    username: !secret router_user
    password: !secret router_password
    interval_seconds: 120
    consider_home: 300
    track_new_devices: yes
```

The device tracker configuration page (http://bit.ly/2s4WPcA) gives more details about what options you can use. The interval_seconds option is the time between scans (2 minutes) and the consider_home option keeps you "at home" even if your devices is not seen for 300 seconds.

Once you restart HA, after the initial discovery is done a new file will be created, called known_devices.yaml. Here you will be able to assign a friendly name and even a picture to a specific device, or have other devices be ignored.

One entry in known_devices.yaml may look like this:

```
aldebaran:
  hide_if_away: false
  mac: 00:1E:06:31:8C:5B
  name: aldebaran
  picture: /local/aldebaran.png
  track: true
  vendor: WIBRAIN
```

Notice that I added a path to local picture which is stored in /home/homeassistant/.homeassistant/www/aldebaran.png. You can create the "www" folder with the following command:

```
 $ sudo mkdir /home/homeassistant/.homeassistant/
www
```

If there are devices which you don't want to monitor, you can set "track: false" in the known_devices.yaml file.

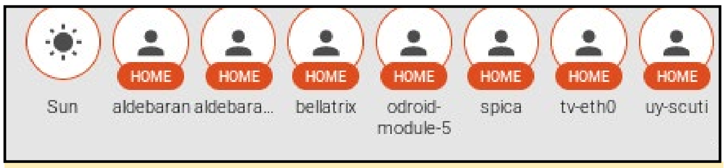
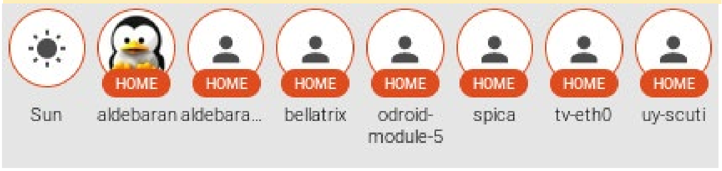


Figure 7 - Initial discovery/ Customized entries



Measuring temperature

A very powerful feature of Home Assistant is the ability to track all sorts of sensors (`http://bit.ly/2cNb4gJ`). We want to monitor a temperature sensor based on the 1 wire protocol, connected locally to the ODROID (`http://bit.ly/2s12ZPx`). Before adding the sensor in HA, make sure it's readable from the command line. You can follow the setup guide on the wiki at `http://bit.ly/2s0zbTp`.

You will need to know the sensor's ID in order to add it to HA:

```
  $ ls /sys/bus/w1/devices/
  28-0516866e14ff  w1_bus_master1
  $ cat /sys/bus/w1/devices/28-0516866e14ff/w1_
slave
  92 01 4b 46 7f ff 0c 10 b5 : crc=b5 YES
  92 01 4b 46 7f ff 0c 10 b5 t=25125
```

Next, you can make the following changes in configuration.yaml and poll the sensor every 5 minutes:

```
sensor:
...
  - platform: onewire
    names:
      28-0516866e14ff: Living room
      scan_interval: '00:05'
```

After restarting HA, the new reading will be visible in the web interface as a badge in the top part of the page.

Sorting the views

You will notice that once you start adding a few components, the web interface starts to get messy with a lot of items scattered everywhere. You can use groups and views to clean up the interface and put related items in their own tab. To understand what needs to be done, let's clear the vocabulary.

Entities are variables which provide data, such as a sensor or switch. Platforms (like dark_sky) usually provide access to multiple entities (min/max temperatures or forecast). You can view a list of entities, their names and their value if you navigate in the web interface under Developer tools -> States (<>) -> Entities.

A group is simply an object that holds a list of entities. Visually, a group is rendered as a panel, or a card. By default the group "group.all_devices" exists and holds the items discovered by a device tracker platform. Groups usually contain a list of entities.

A view is rendered as a separate tab inside Home Assistant. Views are actually groups of groups and differ from regular groups by having the property of "view: yes". You can also add individual entities, as well as groups to a view.

We will group our existing sensors into the following categories:

The first tab is called Home and contains the following groups (it will be internally called default_view, so that it is displayed when you log in):

- **Weather data**
- **Presence information**
- **System information (to show you if there are updates available)**

The second tab is called Media and contains the following groups:

- **Media players**
- **The final tab is called Images and contains:**
- **Webcams**

The configuration looks similar to the list above:

```
group:
  default_view:
    view: yes
    entities:
    - group.weather
    - group.presence
    - group.systeminfo
  media:
    view: yes
    entities:
    - group.mediaplayers
  images:
    view: yes
    entities:
    - camera.observatory_in_spain
    - camera.local_file
  weather:
    name: Weather
    entities:
    - sensor.dark_sky_apparent_temperature
    - sensor.dark_sky_daily_high_temperature
    - sensor.dark_sky_daily_low_temperature
    - sensor.dark_sky_hourly_summary
    - sensor.living_room
  presence:
    name: Presence
    entities:
    - device_tracker.aldebaran
    - device_tracker.nutty
  systeminfo:
    name: System Info
    entities:
    - updater.updater
```

NEW HARDKERNEL LOGO

A SHINY NEW COMMUNITY LOGO FOR OUR FORUM

edited by Rob Roy

Have you noticed something different about the ODROID Forums recently? There is a new ODROID community logo which was designed and contributed by Daniel Mehrwald (@AreaScout), who composited the word "ODROID", a picture of our hardware, and a kernel image.

ODROID
by Hardkernel

WINDOWS X86 COMPATIBILITY LIST FOR ODROID

XU3/XU4 RUNNING EXAGEAR 2.1 DESKTOP

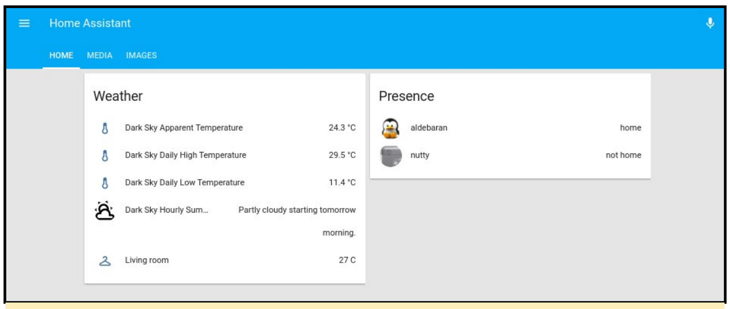
edited by Bruno Doiche

Want to know what we are running lately using our Exagear on our XU3/XU4? Look no further! Here you can look at our most updated list of classic PC gaming that will make your investment worth every penny!

7th Legion - Alien Nations - Age of Wonders - Alpha Centaruri + Alien Crossfire - Anno 1602 - Arcanum: Of Steamworks and Magick Obscura - Balls of Steel - Caesar III - Dune 2000 - Earth 2140 - Homeworld: Cataclysm - KKND2 Krossfire - Larry 7: Love for Sail

Every single game listed here will run perfectly on your ODROID! Enjoy, and for more info go to:

`https://oph.mdrjr.net/meveric/other/ExaGear/`



Weather

Dark Sky Apparent Temperature 24.3 °C
Dark Sky Daily High Temperature 29.5 °C
Dark Sky Daily Low Temperature 11.4 °C
Dark Sky Hourly Sum... Partly cloudy starting tomorrow morning.
Living room 27 C

Presence

aldebaran home
nutty not home

Figure 8 - A cleaner interface with views and groups

```
    mediaplayers:
      name: Media Players
      entities:
      - media_player.mpd_livingroom
      - media_player.kodi_livingroom
```

More details about groups and layout are available in the video at `http://bit.ly/2s5d6xT`.

Updates

Since Home Assistant was not installed via apt-get, you will need to handle updates manually. Before updating, it's best to read the release notes and verify that the update is not breaking any previous configurations, since the configuration for new components sometimes gets redesigned, which means you'll need to redo it. You can get a notification for a new version by using the updater.updater entity which periodically checks for newer versions and can display them inside Home Assistant. Updates are pretty frequent, and you can expect a major version every 2-3 weeks. The update procedure is simple, and details can be found at `http://bit.ly/2s0Kn24`.

```
  $ sudo service homeassistant stop
  $ sudo su -s /bin/bash homeassistant
  $ source /srv/homeassistant/bin/activate
  (homeassistant)$ pip3 install --upgrade homeassis-
tant
  (homeassistant)$ exit
  $ sudo service homeassistant start
```

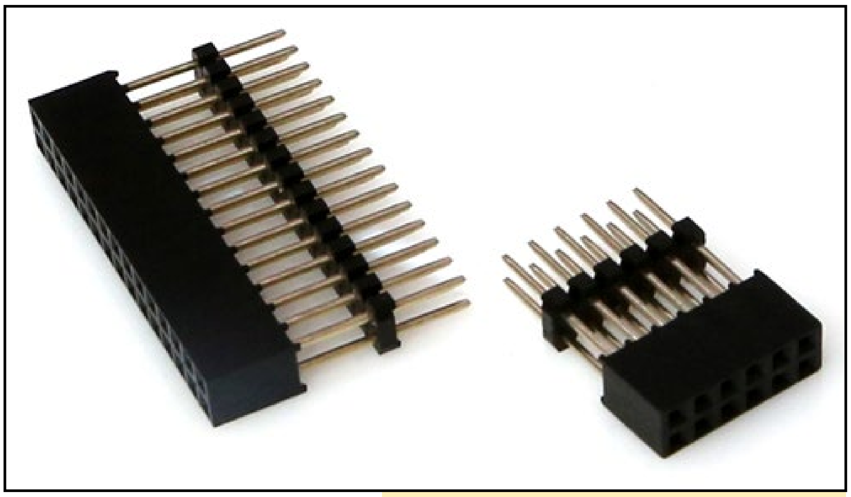
In subsequent articles, we will look at setting up more complex components like a remote relay or an air conditioning unit, setting up automations, and setting up a dashboard. For comments, questions and suggestions, please visit the support thread at `http://bit.ly/2s13GbB`.

ODROID-XU4 HEADER EXTENDERS

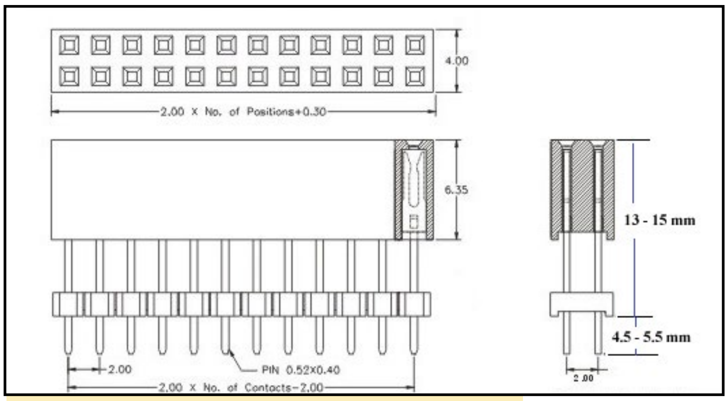
HIGHER SOCKETS FOR YOUR PROJECTS

edited by Rob Roy

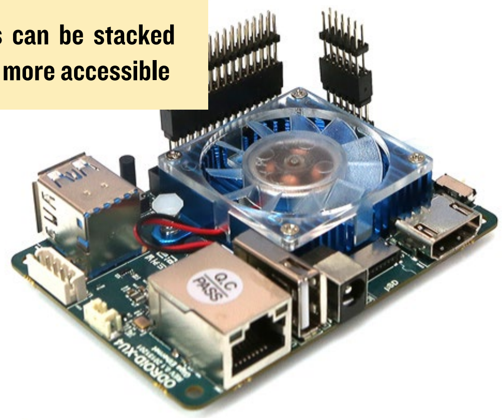
There was a request recently from two of the ODROID forum members for a header extender to increase the height of the 30-pin and 12-pin sockets on the ODROID-XU4 and ODROID-XU4Q. Hardkernel responded by creating a new product, available at `http://bit.ly/2t4Qouj`, that allows multiple shields to be stacked on top of the ODROID-XU4, such as the Shifter Shield (`http://bit.ly/2t4huSJ`).

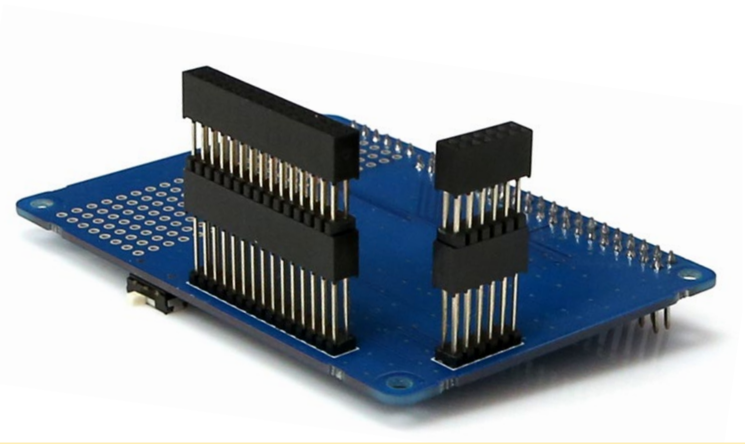

30-pin and 12-pin header extenders

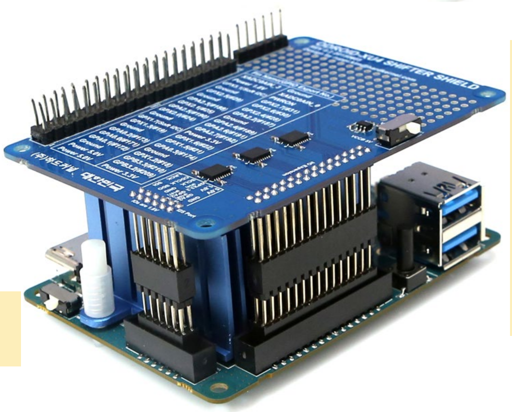
The header extenders can be stacked to make the GPIO pins more accessible



Header extenders technical specifications



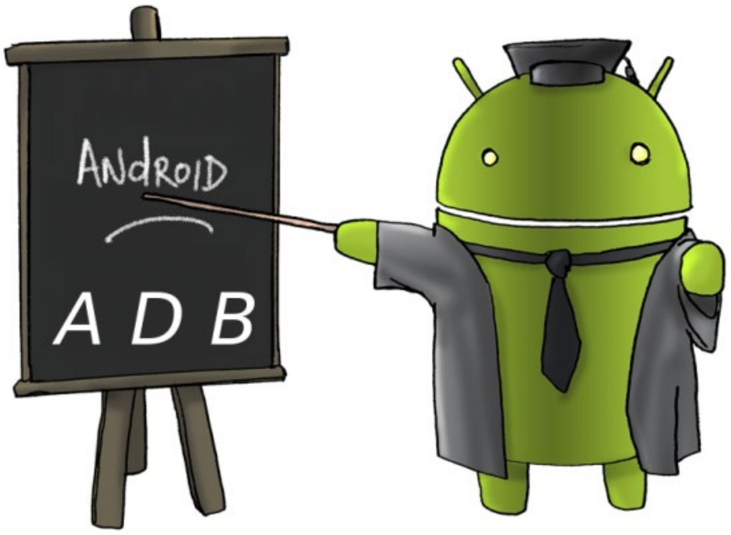

The sockets are useful when you want to make the Shifter Shield taller


The ODROID-XU4Q with header extenders and Shifter Shield installed on top of the blue heatsink

ANDROID DEVELOPMENT

DEEP DIVING INSIDE ADB PART 2

by Nanik Tolaram



This is the second installment of a 2-part series about Android ADB, where we will take an in-depth look at the source code and how it works internally. We will look at the the 2 components of adb that run on the desktop when you run the "adb" command.

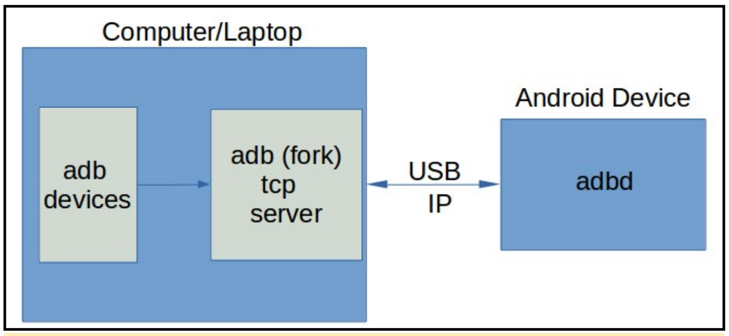


Figure 1 - ADB Architecture

In part 1, we saw the diagram shown in Figure 1, and we are going to go more in detail regarding how the adb runs inside your computer when you connect an Android device to your PC/Laptop. The discussion in this article is based on adb from Android 5.1.1, and in general, the logic should be similar to newer versions of Android with only slight differences in behaviour and code.

Deeper Architecture

As you can see in Figure 2, when you execute adb the first thing it will do is setup a local server with port 5037.

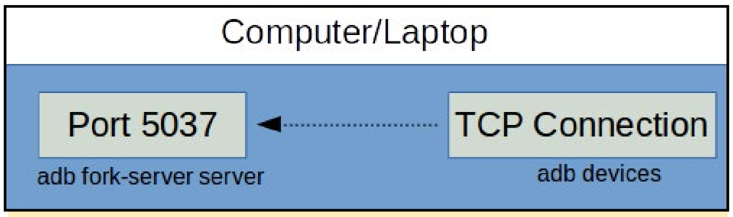


Figure 2 - ADB local server 5037

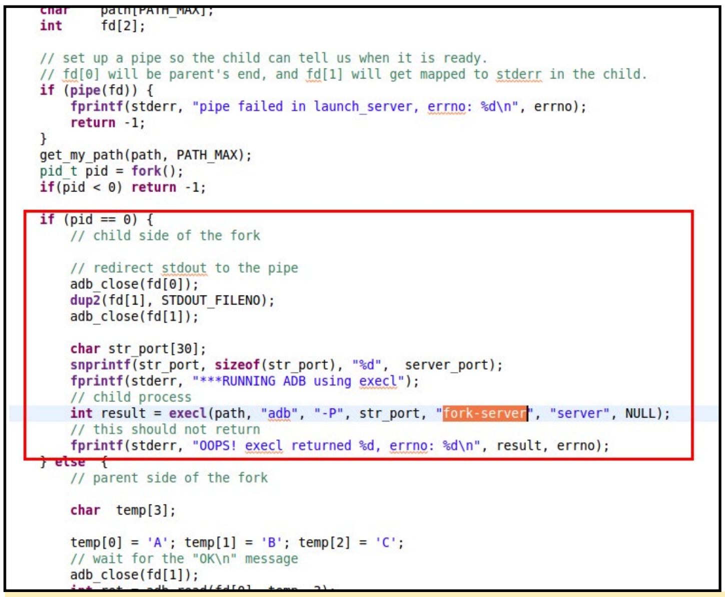
Port 5037 will be used internally by the code that will execute the real command. Let's take a look at an example when we execute a particular command in adb:

```
$ adb devices
```

When adb run the first time, it will spawn another adb process with the following extra parameter:

```
'fork-server server'
```

Figure 3 shows the code that is doing exactly this.



```
char    path[PATH_MAX];
int     fd[2];

// set up a pipe so the child can tell us when it is ready.
// fd[0] will be parent's end, and fd[1] will get mapped to stderr in the child.
if (pipe(fd)) {
    fprintf(stderr, "pipe failed in launch_server, errno: %d\n", errno);
    return -1;
}
get_my_path(path, PATH_MAX);
pid_t pid = fork();
if(pid < 0) return -1;

if (pid == 0) {
    // child side of the fork

    // redirect stdout to the pipe
    adb_close(fd[0]);
    dup2(fd[1], STDOUT_FILENO);
    adb_close(fd[1]);

    char str_port[30];
    snprintf(str_port, sizeof(str_port), "%d",  server_port);
    fprintf(stderr, "***RUNNING ADB using execl");
    // child process
    int result = execl(path, "adb", "-P", str_port, "fork-server", "server", NULL);
    // this should not return
    fprintf(stderr, "OOPS! execl returned %d, errno: %d\n", result, errno);
} else  {
    // parent side of the fork

    char  temp[3];

    temp[0] = 'A'; temp[1] = 'B'; temp[2] = 'C';
    // wait for the "OK\n" message
    adb_close(fd[1]);
    int ret = adb_read(fd[0], temp, 3);
```

Figure 3 - ADB fork-server spawn process

The spawned process will run and and act as a server by listening to local port 5037, while the original process will wait and eventually connect to port 5037 passing the command "devices".

ADB Server

Now that we know that adb will spawn another adb process and run it as server, we are going to take a look what actually happens when it runs as a server. Figure 4 shows the code flow.

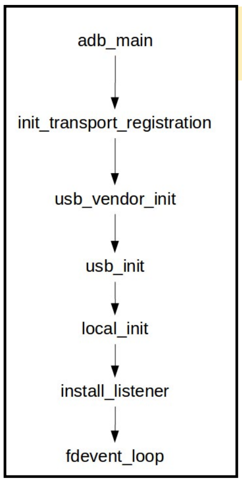
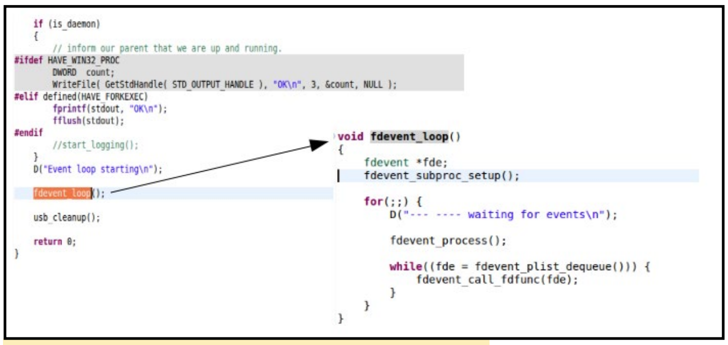


Figure 4 - Code flow running as server

The fdevent_loop shown in Figure 5 is a forever loop that waits for a new event to arrive.

The event management is using an internal object called fdevent which is used in conjunction with synchronous I/O multiplexing that is available as part of the Linux API. The fdevent registration process uses the file descriptor that the code is listening to, and when data is received for that particular file descriptor, the loop will call the registered function inside the function fdevent_call_fdfunc().



```
    if (is_daemon)
    {
        // inform our parent that we are up and running.
#ifdef HAVE_WIN32_PROC
        DWORD  count;
        WriteFile( GetStdHandle( STD_OUTPUT_HANDLE ), "OK\n", 3, &count, NULL );
#elif defined(HAVE_FORKEXEC)
        fprintf(stdout, "OK\n");
        fflush(stdout);
#endif
        //start_logging();
    }
    D("Event loop starting\n");

    fdevent_loop();

    usb_cleanup();

    return 0;
}
```

```
void fdevent_loop()
{
    fdevent *fde;
    fdevent_subproc_setup();

    for(;;) {
        D("--- ---- waiting for events\n");

        fdevent_process();

        while((fde = fdevent_plist_dequeue())) {
            fdevent_call_fdfunc(fde);
        }
    }
}
```

Figure 5 - fdevent_loop

ADB Client

Once the above step of running adb as a server is done, the code will continue to execute by connecting to the local port 5037 and relay our example command 'devices'. The code will wait for the response from the server and will print out the result once it's completed. Table 1 shows some of the most used functions related to fdevent relevant to adb.

Transport Registration

ADB supports connection via USB and TCP which makes developer's lives so much easier. When the server code is running, there is background thread that listens for any incoming USB connections. Once a new USB device is detected, it will use fdevent_install to install a new function for the file descriptor in order to perform communication between the device and the local server.

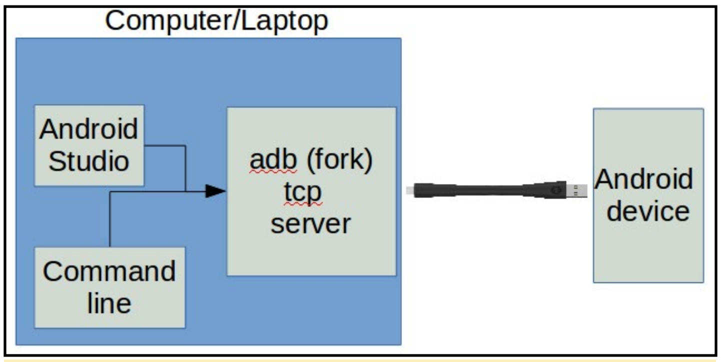


Figure 7 - Transport Registration

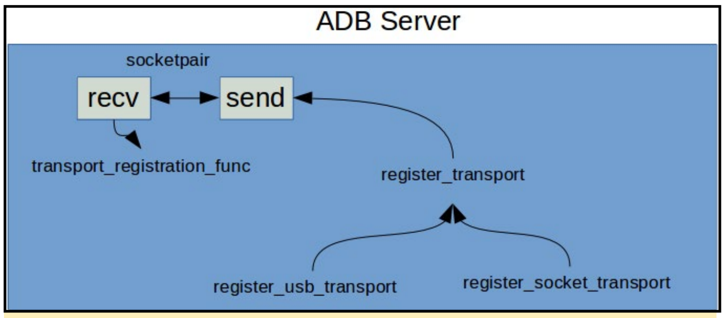
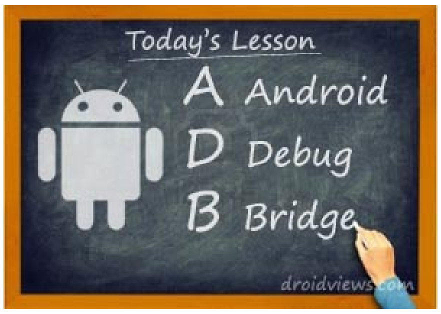


Figure 6 - USB Connection to Android Device

Figure 7 outlines the registration process inside ADB for USB and TCP. Internally, there is a socketpair that is used to communicate when a new Android device is detected. This is important to allow adb handle multiple Android devices at the same time. The function register_usb_transport is responsible for USB connections, while register_socket_transport is responsible for network (TCP) connections. Once the connection is detected and handled by adb, it creates a separate thread to handle communication between itself and the device.

fdevent_install	Register function as a callback for a particular file descriptor
fdevent_set	Flag the file descriptor to wait for reading, writing or error handling
fdevent_add	Add flag to the file descriptor
fdevent_del	Delete flag from the file descriptor
fdevent_loop	Loop to process all incoming events

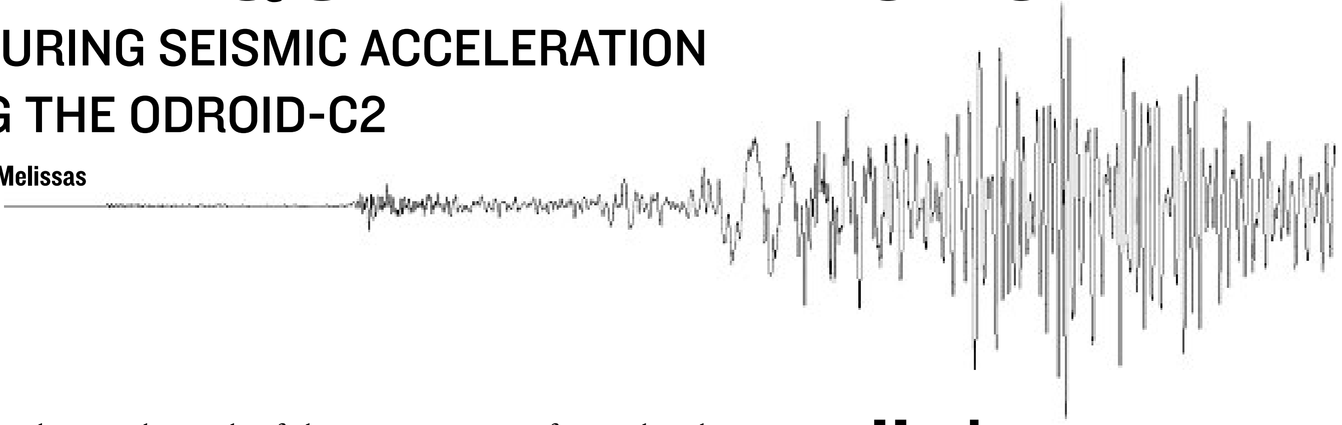
Table 1 - Some of the available fdevent functions



SEISMOGRAPH EARTHQUAKE DETECTOR

MEASURING SEISMIC ACCELERATION USING THE ODROID-C2

by Miltiadis Melissas



Earthquakes are the result of the sudden release of energy in the Earth's lithosphere that creates seismic waves (Figure 1). Earthquakes range in magnitude from those so weak they cannot be felt, to those violent enough to toss people around and destroy entire cities. Seismicity, or seismic activity, refers to the frequency, type, and size of earthquakes experienced over a period of time in a given area (http://bit.ly/1WBShEL).

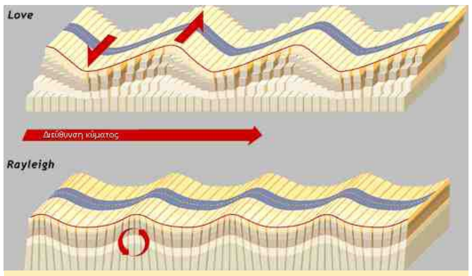


Figure 1 - Diagram of a seismic event

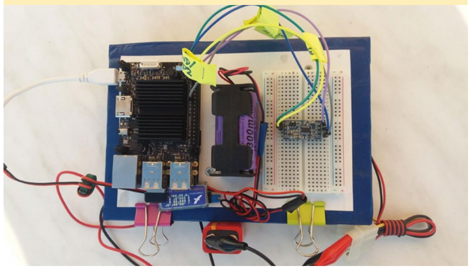
In this project, we will build an earthquake detector seismographic device using the ODROID-C2 together with an accelerometer (MMA7455) for detecting and measuring the acceleration gravity caused by those seismic waves. Our seismograph will be capable of:

- Measuring an earthquake's magnitude in grams (g)
- Detecting and measuring an earthquake's acceleration in three different axes (x,y,z)
- Recording the time and the duration of an earthquake
- Representing all of the values above with real-time graphics
- Reporting GPS coordinates on Google Maps
- Posting to Twitter when seismic acceleration reaches a certain limit--usually 0.3g--notifying citizens of the time and magnitude where the earthquake was recorded

This device provides numerous advantages, such as:

- No mechanical parts, reducing friction and increasing accuracy
- Open source availability
- Inexpensive, making it ideal for building a network of seismographs
- Can be programmed and controlled remotely through the web
- The seismograph's reports are accessible all over the world, from any device with an Internet connection

Photo 1 - ODROID-C2 as a seismograph



Hardware

You will need all of the usual ODROID-C2 accessories:

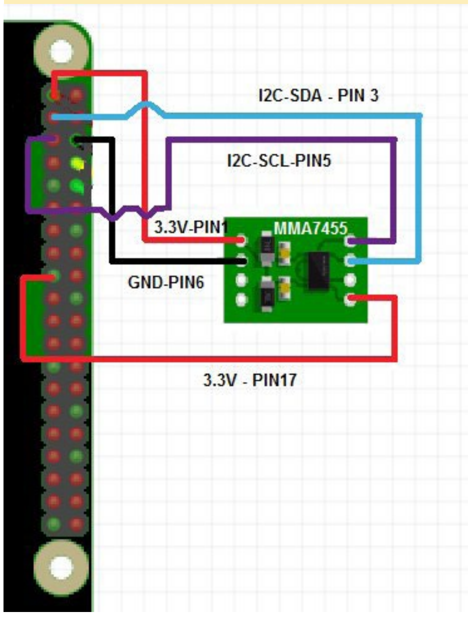
- ODROID-C2
- MicroSD card with the latest Ubuntu 16.04 provided by Hardkernel (http://bit.ly/2rDOCfn)
- WiringPi library for controlling the GPIOs of an ODROID-C2 running on Ubuntu 16.04. Instructions from Hardkernel on how to install the library can be found at http://bit.ly/1NsrlU9
- Keyboard
- Screen
- HDMI cable
- Micro USB power or, even better, a power supply provided by Hardkernel (http://bit.ly/1X0bgdt)
- Optional: Power bank with UBEC (3A max, 5V) if you want to operate the device autonomously (see Photo 1). Hardkernel provides a better solution with a UPS3 specifically designed for ODROID-C2. You can purchase this supply from their store at this link: http://bit.ly/2l1rE25. The UPS3 is a good choice, as it gives the detector the ability to operate autonomously with greater stability and duration
- Ethernet cable or usb wifi dongle

and Internet router

- A 90mm PC fan, to be modified for use as a regression plate
- The C Tinkering Kit on Ubuntu which can be purchased from Hardkernel (http://bit.ly/1NsrlU9)
- MMA7455 accelerometer module, which can be found from various places, including eBay

For the wiring, please follow our schematic in Figure 2. You will need 3.3V for the digital and the analog part of the accelerometer, so connect GPIO_1 to Pin 1 of the accelerometer, which will provides the 3.3V. Your ground wire should connect Pin 6 on the GPIO to Pin 2 on accelerometer. Pin 7, the ChipSelect (CS), connected to Pin 17 of the ODROID-C2's GPIO, adding as another 3.3V are needed for this purpose (active low for SPI otherwise, as in this case, I2C). There are 2 important wires left for the communication: the SDA that provides the I2C serial data and the SCL that provides the I2C serial clock. The SDA is on Pin 13 on accelerometer and is connected on GPIO Pin 3 of ODROID. Last but not leas,t the SCL is on Pin 14 and is connected on GPIO Pin 5 of ODROID. Please refer again both to our schematic below (Figure 2) and to Hardkernel's excellent 40-pin layout for ODROID-C2 (http://bit.ly/2aXAlmt) for doing the wiring correctly.

Figure 2 - Wiring schematic diagram



Now that we have our hardware ready, let's see how we can establish a communication between the ODROID-C2 and the MMA7455 accelerometer using the I2C protocol.

I2C communication

All commands are entered in a terminal window or via SSH. First, you'll need to update ODROID-C2 to ensure all the latest packages are installed:

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

Then you will need to reboot the ODROID-C2:

```
$ sudo reboot
```

You will need to install SMBus and I2C-Tools since the MMA7455 accelerometer uses this protocol to communicate with the ODROID-C2. The System Management Bus (SMBus) is a simple, single-ended, two-wire bus for lightweight communication. It is most commonly found in computer motherboards for communicating with the power source (http://bit.ly/2rAWhuU). The I2C protocol is a multi-master, multi-slave, packet switched, single-ended, serial computer bus invented by Philips Semiconductor (now NXP Semiconductors). It is typically used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication (http://bit.ly/2qGiYP4).

Getting back to our guide, once you have logged into your ODROID-C2 from the command line, run the following command to install Python-SMBus and I2C-Tools:

```
$ sudo apt-get install python-
smbus
$ sudo apt-get install i2c tools
```

Set the ODROID-C2 to load the I2C driver:

```
$ modprobe ami-i2c
```

Set the ODROID-C2 to start I2C automatically at boot by editing /etc/modules:

```
$ sudo nano /etc/modules
```

Use your cursor keys to move to the last line. Add a new line and then add:

```
$ i2c-dev
```

Press return, then add:

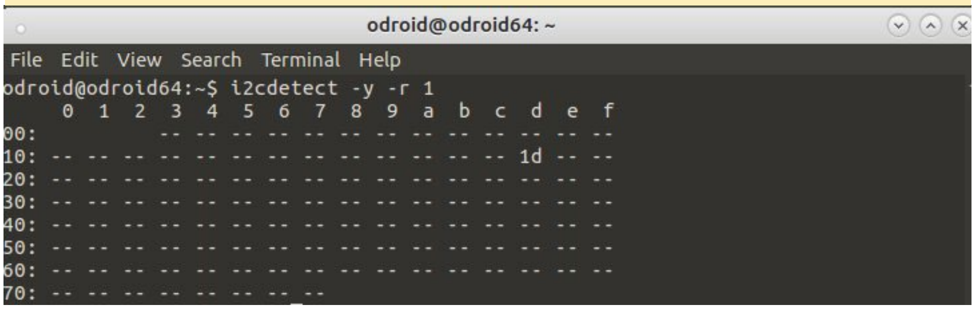
```
$ aml_i2c
```

Save your changes and exit the nano editor. To avoid having to run the I2C tools at root add the "ODROID" user to the I2C group:

```
$ sudo adduser Odroid i2c
```

Next reboot the ODROID-C2:

Figure 3 - Sample accelerometer output



```
odroid@odroid64:~$ i2cdetect -y -r 1
     0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:          -- -- -- -- -- -- -- -- -- -- -- -- --
10: -- -- -- -- -- -- -- -- -- -- -- -- -- 1d -- --
20: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
30: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
40: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
50: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
60: -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --
70: -- -- -- -- -- -- -- --
```

```
$ sudo reboot
```

Once your ODROID-C2 has been rebooted, you will have I2C support. You can check for connected I2C devices with the following command:

```
$ sudo i2cdetect -y -r 1
```

If 1d displays on line 10 under column D, this means the accelerometer is communicating with ODROID-C2 and working properly (Figure 3). More details may be found at http://bit.ly/2qCQM1s.

Python software

We will present the code in chunks, in order to be better understood. First, import the necessary modules:

```
$ import wiringpi2 as odroid
$ import time
$ from import sleep
```

Next, set up the WiringPI library:

```
$ odroid.wiringPiSetup()
$ DEBUG = 1
```

Setup the API and posting data delay for the ThingSpeak platform (https://thingspeak.com). Please refer to the section below on how to register and use the ThingSpeak platform.

```
$ myAPI - "XXXXXXXXXXXXXXX" #
This is the API key provided by
ThingSpeak
$ myDelay - 1 # How many seconds
between posting data
```

The next piece of code is for controlling the SMBus for the communication between the accelerometer and the ODROID-C2 via the I2C protocol:

```
$ class Accel():
$ myBus = 1
$ b = smbus.SMBus(myBus)
```

Define a class called Accel for calibrating the sensor and for reading values of acceleration on the three axes (x,y,z). Those values represent the measurements of the acceleration of gravity when the seismic waves "hit" the device:

```
$ def setUp(self):
      $ self.b.write_byte_
data(0x1D,0x16,0x55) - Set up the
Mode (sensitivity=2g)
      $ self.b.write_byte_
data(0x1D,0x10,0) - Calibrate
sensor
      $ self.b.write_byte_
data(0x1D,0x11,0) - Calibrate
sensor
      $ self.b.write_byte_
data(0x1D,0x12,0) - Calibrate
sensor
      $ self.b.write_byte_
data(0x1D,0x13,0) - Calibrate
sensor
      $ self.b.write_byte_
data(0x1D,0x14,0) - Calibrate
sensor
      $ self.b.write_byte_
data(0x1D,0x15,0) - Calibrate
sensor
$ def getValueX(self):
$ return self.b.read_byte_
data(0x1D,0x06)
$ def getValueY(self):
$ return self.b.read_byte_
data(0x1D,0x07)
$ def getValueZ(self):
$ return self.b.read_byte_
data(0x1D,0x08)
```

Create the object MMA7455:

```
$ MMA7455 = Accel()
$ MMA7455.setUP()
```

For the next piece of code, you'll need to use a little bit of math to transform the raw values of the accelerometer to more meaningful values, using the accelerometer of a smartphone for the calibration. You might have to work a little bit more with your own smartphone, so the working conditions of your device for the calibration of those formulas may vary as a result.

```
$ def get SensorData():
      $ x = MMA7455.getValueX()
      $ y = MMA7455.getValueY()
      $ z = MMA7455.getValueZ ()
      $ x2=(((x-115)%256)-(128))/
float(100) - Calibrate values x
      $
y2=(((y-226+128)%256)-128)/
float(100) - Calibrate values y
$ z2=(((z-126+128)%256)-128)/
float(10>) - Calibrate values z
$ mag=(math.
sqrt(x2*x2+y2*y2+z2*z2))
$ time.sleep(01) - Sampling every
1s
$ return (str(x2), str(y2),
str(z2), str(mag))
```

Below you will find the main function for posting data to the ThingSpeak platform and visualizing the data as the earthquake waves "hit" the device. Remember that the earthquake detector is capable of recording vibrations on the X,Y,Z axes as well.

```
$ # main() function
$ def main():
```

Finally, print the results:

```
$ print "starting…"

$ baseURL = "https://api.thing-
speak.com/update?api_key=%s" %
myAPI
$ print (baseURL)

$ while True:
$ try:
$ x2, y2, z2, mag = getSensor-
Data()

$ f = urlib2.urlopen(baseURL +
"&field1=%s&field2=%s&field3=%s&fiel
d4=%s" \
% (x2, y2, z2, mag))
```

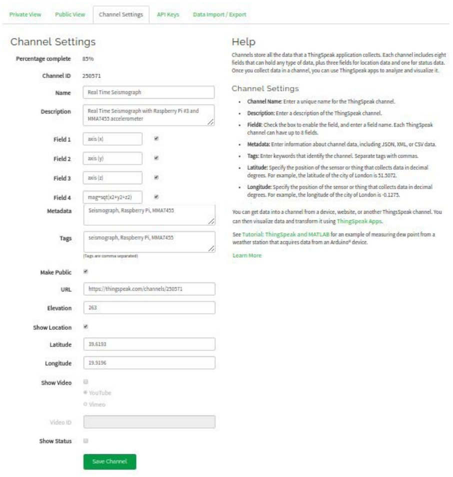
The rest is pure functional code:

```
$ sleep(int(myDelay))
      $ except:
      $ print "exiting."
      $ break
$ # call main
$ if_name_=="_main_":
      $ main()
```

ThingSpeak platform

According to its developers at http://bit.ly/2qrCMoE, ThingSpeak is an open source Internet of Things (IoT) application and API to store and retrieve data from things using the HTTP protocol over the Internet or via a Local Area Network. ThingSpeak enables the creation of sensor logging applications, location tracking applications, and a social network of things with status updates.

After signing up to the service (the sign-up procedure is self-explanatory), login to the ThingSpeak platform with your credentials. You will need to create a "New Channel" for visualizing your seismographic data. From the startup page, click "My Channels" and then "New Channel". Please refer to Figure 4 below for how to set up your channel.



Private View | Public View | Channel Settings | API Keys | Data Import / Export

Channel Settings

Percentage complete: 85%
Channel ID: 250571
Name: Real Time Seismograph
Description: Real Time Seismograph with Raspberry Pi #3 and MMA7455 accelerometer
Field 1: axis (x) ✓
Field 2: axis (y) ✓
Field 3: axis (z) ✓
Field 4: mag=sqt(x2+y2+z2) ✓
Metadata: Seismograph, Raspberry Pi, MMA7455
Tags: seismograph, Raspberry Pi, MMA7455
(Tags are comma separated)
Make Public: ✓
URL: https://thingspeak.com/channels/250571
Elevation: 263
Show Location: ✓
Latitude: 39.6193
Longitude: 19.9196
Show Video: ☐
YouTube
Vimeo
Video ID:
Show Status: ☐
Save Channel

Help

Channels store all the data that a ThingSpeak application collects. Each channel includes eight fields that can hold any type of data, plus three fields for location data and one for status data. Once you collect data in a channel, you can use ThingSpeak apps to analyze and visualize it.

Channel Settings

- **Channel Name:** Enter a unique name for the ThingSpeak channel.
- **Description:** Enter a description of the ThingSpeak channel.
- **Field#:** Check the box to enable the field, and enter a field name. Each ThingSpeak channel can have up to 8 fields.
- **Metadata:** Enter information about channel data, including JSON, XML, or CSV data.
- **Tags:** Enter keywords that identify the channel. Separate tags with commas.
- **Latitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the latitude of the city of London is 51.5072.
- **Longitude:** Specify the position of the sensor or thing that collects data in decimal degrees. For example, the longitude of the city of London is -0.1275.

You can get data into a channel from a device, website, or another ThingsSpeak channel. You can then visualize data and transform it using ThingSpeak Apps.

See Tutorial: ThingSpeak and MATLAB for an example of measuring dew point from a weather station that acquires data from an Arduino® device.

Learn More

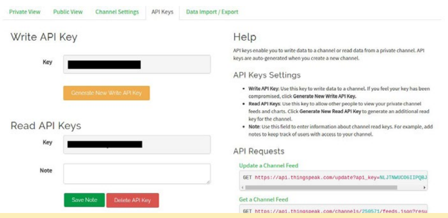
Figure 4 - How to set up your channel

The "Name" and the "Description" of the channel is obvious. Fill in those fields with something descriptive. What matters most are the four fields we defined (ie. Field 1, 2, 3, and 4). Understandably, those fields represent the acceleration gravity on the three axes (x,y,z) plus one, Field 4, which represents the magnitude of the seismic acceleration. Fill in the respective fields with the appropriate descriptions. Lastly put the "Latitude", "Longitude" and the "Elevation" of your device for a proper representation of the spot on Google Maps (fifth graph at the bottom). This is a pseudo GPS function, but it's better than nothing!

As soon as you save your Channel a new "ID Channel" will be assigned to it. This "ID Channel" also forms your web address, so don't underestimate it. For example, our "ID Channel" is "25071," so our data are accessible via this Internet address: http://www.thingspeak.com/25071.

Of course this address is accessible from any computer, tablet, laptop, or smartphone, which is a very powerful characteristic of this device.

Additionally, you will need is an API Key(s) for accessing the platform through our code. The API Key(s) are accessible via the "API Keys" tab at the top of the page. As soon as you click on it the following page appears (Figure 5).



Private View | Public View | Channel Settings | API Keys | Data Import / Export

Write API Key

Key: [redacted]

Generate New Write API Key

Read API Keys

Key: [redacted]

Note:

Save Note | Delete API Key

Help

API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.

API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key.**
- **Read API Keys:** Use this key to allow other people to view your private channel feeds and charts. Click **Generate New Read API Key** to generate an additional read key for the channel.
- **Note:** Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.

API Requests

Update a Channel Feed

```
GET https://api.thingspeak.com/update?api_key=NLJTNWUCO6IIPQBJ
```

Get a Channel Feed

```
GET https://api.thingspeak.com/channels/250571/feeds.json?resu
```

Figure 5 - API Keys web interface

The most important field is the "Write API Key," as this will write or post data to the ThingSpeak platform. Recall the lines of your python code on the "seismos.py" a little bit earlier, lines that make use of this API Write key:

```
$ myAPI = "XXXXXXXXXXXXXXXX" # This
is the API key provided by the
ThingSpeak platform.
$ ---
$ baseURL = "https://api.thingspeak.
com/update?api_key=%s" % myAPI
```

That's all you'll need regarding ThingSpeak and setting up the platform to depict your data on the device.

Simulating seismic waves

How should you test your earthquake detector? Of course, you're not going to want to sit around and wait for an earthquake! You'll want to build a regression mechanism in order to produce vibrations to the device at will. This so-called "regression plate" can easily be built by using a PC cooling fan with some minor modifications. For ours, we took a 90mm PC cooling fan and cut three of its seven rotating blades. By doing so, we created an "offset" or a cam rotating mechanism that vibrates (Photo 3). As a final step, we attached this improvised mechanism below the earthquake detector's plate.

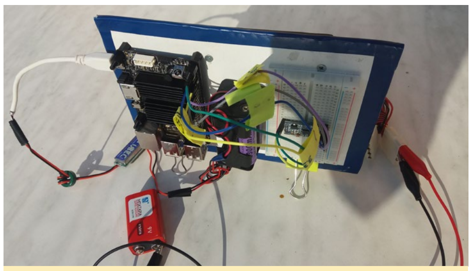


Photo 2 - Side view of the seismograph

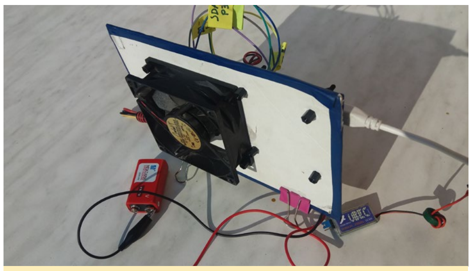


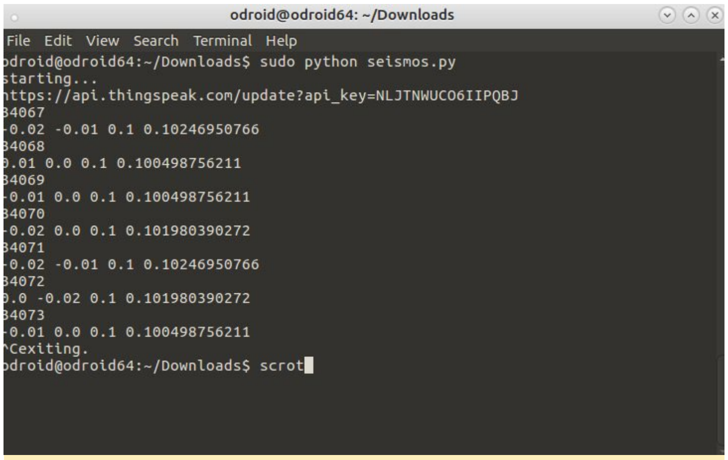
Photo 3 - Opposite side view

Testing the detector

Type the following into a Terminal window to test the seismograph:

```
$ sudo python seismo.py
```

Siesmos.py immediately posts data to ThingSpeak, while also reporting data locally to the terminal (Figure 6).


odroid@odroid64: ~/Downloads
File Edit View Search Terminal Help

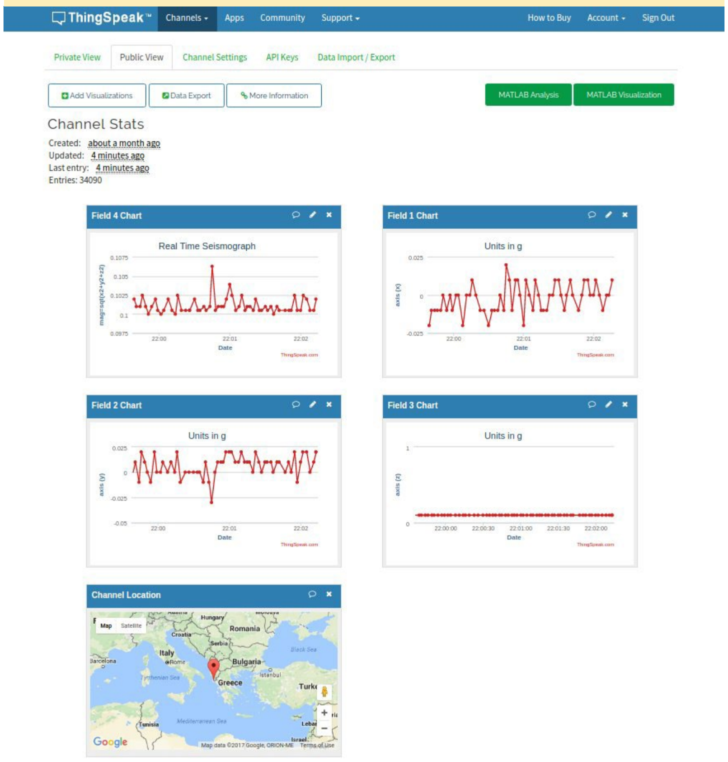
```
odroid@odroid64:~/Downloads$ sudo python seismos.py
starting...
https://api.thingspeak.com/update?api_key=NLJTNWUC06IIPQBJ
34067
-0.02 -0.01 0.1 0.10246950766
34068
0.01 0.0 0.1 0.100498756211
34069
-0.01 0.0 0.1 0.100498756211
34070
-0.02 0.0 0.1 0.101980390272
34071
-0.02 -0.01 0.1 0.10246950766
34072
0.0 -0.02 0.1 0.101980390272
34073
-0.01 0.0 0.1 0.100498756211
^Cexiting.
odroid@odroid64:~/Downloads$ scrot
```

Figure 6 - Seismos.py output to ThingSpeak

The acceleration gravity values are reported every one second and then uploaded to the ThingSpeak platform, which transforms those values to graphs. Altogether, there are four graphs, the first one of which is most important as it represents the seismic acceleration by using the values of all the other graphs (three in total). In other words, this is the magnitude of the seismic acceleration (Mg) as a result of this mathematical formula (Mg=). The coolest part of this project is that those graphs are accessible via the ThingSpeak platform (at http://bit.ly/2s0mDqM in our case) from any web-connected device, such as computers, tablets, laptops, and even smartphones. You might notice some minor vibrations ranging from -0.3g to +0.3g before starting to shake the device using your regression fan, but those vibrations are purely noise from your surroundings (Figure 7).

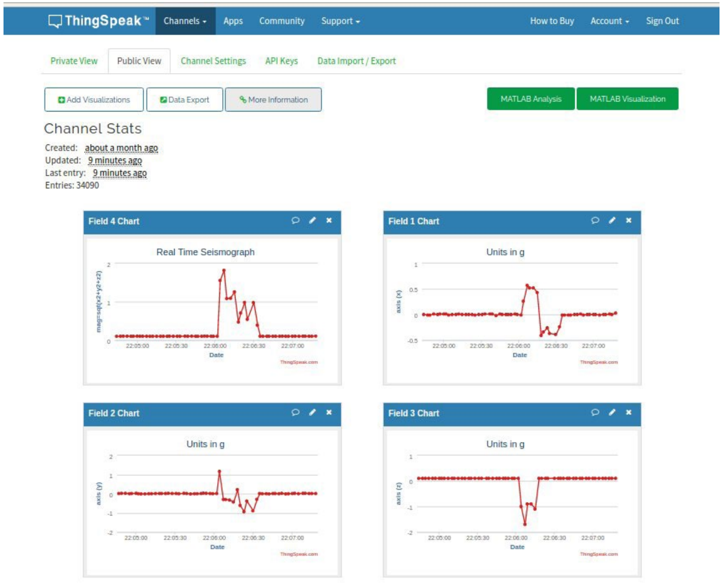
The last graph presents the GPS coordinates of the device

Figure 7 - Vibration graphs from the seismograph



Channel Stats
Created: about a month ago
Updated: 4 minutes ago
Last entry: 4 minutes ago
Entries: 34090

on Google Maps. This is not the epicenter of the earthquake, but rather the place where your seismograph device is located. It takes at least three of these devices to determine the epicenter of the earthquake with any accuracy, but that is far beyond the scope of this project. For the time being, try shaking the detector using the regression fan by connecting it to a battery. Can you spot the difference (Figure 8)?



Channel Stats
Created: about a month ago
Updated: 9 minutes ago
Last entry: 9 minutes ago
Entries: 34090

Figure 8 - Sample output from shaking the seismic detector

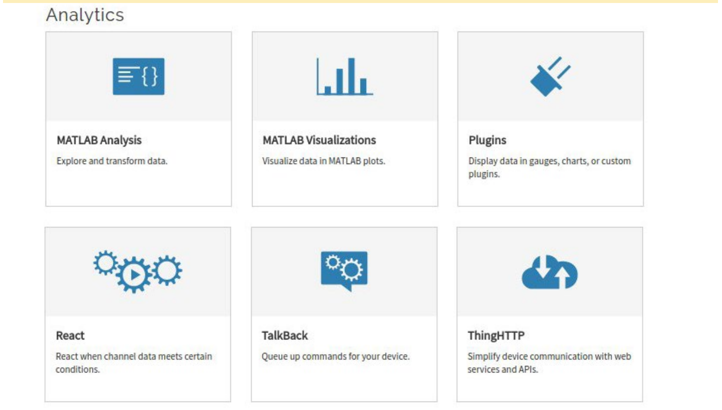
You'll see the magnitude of the seismic acceleration shows higher values than before, much higher, as the simulation of the earthquake waves hit the detector produces your seismograph.

Tweet, tweet, tweet

Now that we have the earthquake detector up and running, let's talk about how to use the detector to inform the public when and if such a catastrophic event happens (we hope not). This part is rather easy once you decide what the baseline value is that should trigger the tweets to be sent.

On the ThingSpeak platform, from the menu at the top of the page, choose "Apps", and then "Actions", and finally from there, "Reacts". You're going want the device to react every

Figure 9 - Setting the parameters for the seismic detector



Analytics

MATLAB Analysis
Explore and transform data.

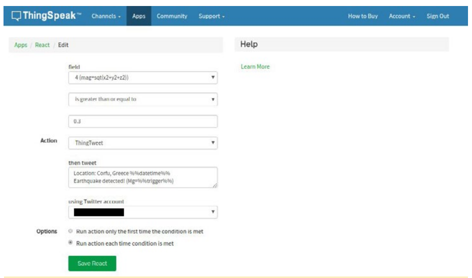
MATLAB Visualizations
Visualize data in MATLAB plots.

Plugins
Display data in gauges, charts, or custom plugins.

React
React when channel data meets certain conditions.

TalkBack
Queue up commands for your device.

ThingHTTP
Simplify device communication with web services and APIs.

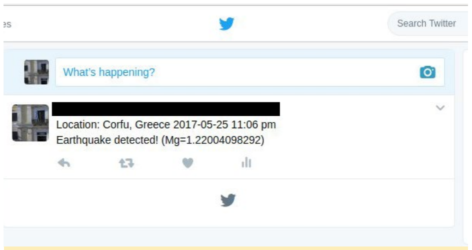


field: 4 (mag=sqt(x2+y2+z2))
is greater than or equal to
0.3
Action: ThingTweet
then tweet: Location: Corfu, Greece %%datetime%% Earthquake detected! (Mg=%%trigger%%)
using Twitter account: [redacted]
Options: ○ Run action only the first time the condition is met
● Run action each time condition is met

Figure 10

time the seismic acceleration is above the limit you"ve decided on. Click "New React" and set the parameters for this event, referring to Figure 9.

These things are more or less self-explanatory but, just in case you need some clarification, pay attention to the "Condition" field. This is the most criti-



[redacted]
Location: Corfu, Greece 2017-05-25 11:06 pm
Earthquake detected! (Mg=1.22004098292)

Figure 11

cal part of this app as this is where you will set your baseline limit. For example, in this case we set the seismic acceleration (formula) to greater than or equal to 0.3 (our limit). Since this formula is represented by the 4th graph on the 4th field, declare it so (Figure 10).

For the "Action", choose "ThingTweet" and fill in the field "then tweet" with the text you want to include in all of your tweets (e.g. Location: Corfu, Greece %%datetime%%, Earthquake detected! (Mg=%%trigger%%). Finally, set the option "Run action each time condition is met" as you'll want your "React" to be triggered each time the condition is met. The results are shown on our Twitter stream (Figure 11).

Conclusion

Please feel free to make any alterations or add extra capabilities to our earthquake detector, as this device will undoubtedly find some technical use in the scientific field.

GOGS

A GITHUB/GITLAB ALTERNATIVE

by @redrocket

Gogs is a self-hosted Git service. The goal of the project is to make the easiest, fastest, and most efficient way of setting up a self-hosted Git service. With Go, this can be done with an independent binary distribution across all platforms that Go supports, including Linux, Mac OS X, Windows and ARM. This article describes the steps necessary to install Gogs on an ODROID-C2 running Ubuntu Mate, Apache2, MySQL, and PHP 7.0. Make sure to backup all files before attempting to install Gogs!

Compilation

We need to compile the source ourselves, because I wasn't able to run the linux armv5_x64 and RasPi armv6 images provided on the Gogs website. First, create a folder in your home directory where we're going to download the sources, then navigate to that folder:

```
$ mkdir gogs_sources
$ cd gogs_sources
```

Next, install Git and the Go language:

```
$ sudo apt-get install git -y
$ sudo apt-get install golang -y
```

Using Go, download Gogs along with its dependencies:

```
$ go get -u github.com/gogits/
gogs
```

Copy and paste each line below into a Terminal window, where $USER is your Linux username:

```
$ echo 'export GOPATH=/
home/$USER/go' >> $HOME/.bashrc
$ echo 'export PATH=$PATH:GOPATH/
bin' >> $HOME/.bashrc
$ source $HOME/.bashrc
```

Compile the source code:

```
$ cd src/github.com/gogits/gogs
$ go build
```

If no errors occur during compilation, run the following command to test the build:

```
$ ./gogs web
```

If no errors are shown, press Ctrl+C to stop the Gogs server. Move everything to its designated folder:

```
$ cp gogs ~/gogs/
$ cp -a templates/ ~/gogs/
$ cp -a scripts/ ~/gogs/
$ cp -a public/ ~/gogs/
```

According to the documentation, version 0.6.0 and above allows you to create an app.ini configuration file, which must be located in ~/gogs/custom/conf/app.ini:

```
$ pico ~/gogs/custom/conf/app.ini
```

Paste the following configuration snippet into the file:

```
[repository]
ROOT = /path/to/repos-folder

[database]
PASSWD = `root`
```

Do not worry about the PASSWD line. For now, enter the root password, and later when you're installing gogs and create a designated user/pass pair that line will be changed. Save and close the file by pressing Ctrl+X -> Y -> Enter.

Preparing MySQL

To prepare MySQL for the Gogs installation, I used PHPMyAdmin to access MySQL and run the /scripts/mysql.sql file. To do so, log in via PHPMyAdmin, then select "Import" and select the file /home/$USER/gogs/scripts/mysql.sql to be imported. I strongly suggest creating a designated user for the Gogs database, and later use that username/password pair to finish installation of Gogs by using the DB -> Privileges -> Add user account option.

Installation

In a Terminal window, navigate to the Gogs folder and run it:

```
$ cd ~/gogs
$ ./gogs web
```

Open a browser and navigate to http://localhost:3000. Follow the on-screen instructions to finish the installation. On the installation screen, pay particular attention to the user field. By default it's "git", but if you added a different user, you have to change that. At this point you should have a git server running on your internal network. If you want it to be accessible from outside your home network, continue reading.

If you're trying to access it from outside your network, don't forget to change the Application URL to designated one, such as http://git.domain.com. Earlier, I stated that my ODROID-C2 has Apache installed, which I need in order to allow my customers to see the work in progress. So, if we want our Gogs server to be accessible from the Internet, we should combine Apache with a proxy module.

Preparing Apache

To set up Apache to forward the request from the Internet to the Gogs server, let's first check whether we have proxy* modules installed on our Apache server:

```
$ apache2ctl -M
```

At the end of this command, there will be question about which modules you want to install, and in brackets, it will be stated that wildcards are allowed. We can simply type the following and press the Enter key:

```
proxy*
```

Next, create the git.conf file:

```
$ sudo pico /etc/apache2/\
sites-available/git.conf
```

Paste the following configuration snippet into the file, making sure to replace $USER with your Linux username:

```
<VirtualHost *:80>
  ServerAdmin webmaster@localhost
  DocumentRoot /home/$USER/gogs/
public
  ProxyPreserveHost On
  ProxyPass / http://local-
host:3000/
  ProxyPassReverse / http://loc-
alhost:3000/
  ServerName git.domain.com
  ServerAlias git.domain.com
</VirtualHost>
```

Save and close the file by pressing Ctrl+X -> Y -> Enter.

Next, let's enable the new configuration, then restart the Apache service so that the new configuration will take effect:

```
$ sudo a2ensite git.conf
$ sudo service apache2 restart
```

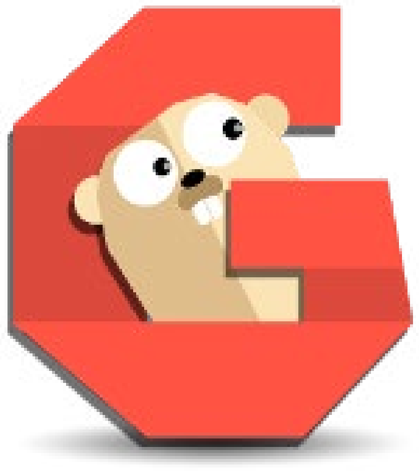
Say a prayer, and when you open http://git.domain.com, you should see the Gogs home screen. If you see an error 503 or something else, you might want to verify whether the Gogs server is running. If not, start it from the console:

```
$ cd ~/gogs
$ ./gogs web
```

Finally, enable gogs.service so that every time you restart your C2, Gogs will run automatically:

```
$ sudo cp ~/gogs/script/systemd/
gogs.service /etc/systemd/system/
$ sudo systemctl enable gogs
$ sudo systemctl start gogs
```

For comments, questions and suggestions, please visit the original thread at http://bit.ly/2sw41jz.



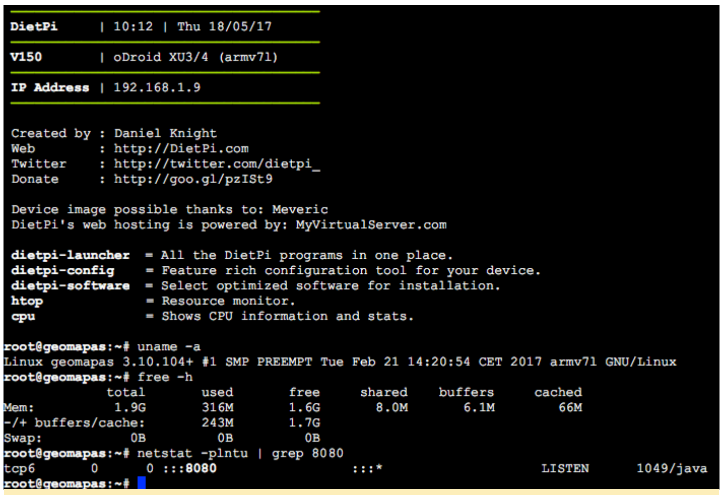
ODROID-XU4 AS A MAP SERVER

A GUIDE TO USING GEOSERVER

by José Cerrejón

I'm going to start a new project to serve map layers to some farming clients. These farmers are going to let us study their land and store this information in the cloud, so that it is always online and ready for access.

It is a fascinating project, and I initially thought about using a Raspberry Pi but its 10/100 Ethernet and 1GB of RAM was insufficient for our purposes, so we aimed for a solution based on the ODROID-XU4 which fit perfectly to our needs. We had the hardware already, albeit forgotten in a drawer. In this article, I will tell you about my adventures in setting up an ODROID-XU4 using DietPi as an operating system and Geo Server as a development environment.



```
DietPi      | 10:12 | Thu 18/05/17
V150        | oDroid XU3/4 (armv7l)
IP Address  | 192.168.1.9

Created by : Daniel Knight
Web        : http://DietPi.com
Twitter    : http://twitter.com/dietpi_
Donate     : http://goo.gl/pzISt9

Device image possible thanks to: Meveric
DietPi's web hosting is powered by: MyVirtualServer.com

dietpi-launcher   = All the DietPi programs in one place.
dietpi-config     = Feature rich configuration tool for your device.
dietpi-software   = Select optimized software for installation.
htop              = Resource monitor.
cpu               = Shows CPU information and stats.

root@geomapas:~# uname -a
Linux geomapas 3.10.104+ #1 SMP PREEMPT Tue Feb 21 14:20:54 CET 2017 armv7l GNU/Linux
root@geomapas:~# free -h
             total       used       free     shared    buffers     cached
Mem:          1.9G       316M       1.6G       8.0M       6.1M        66M
-/+ buffers/cache:       243M       1.7G
Swap:           0B         0B         0B
root@geomapas:~# netstat -plntu | grep 8080
tcp6       0      0 :::8080                 :::*                    LISTEN      1049/java
root@geomapas:~#
```

Figure 1 - DietPi running

DietPi

Using Etcher, I installed the latest version of DietPi, which was downloaded from their site. The installation process was easy, and we had no problems at all. The operating system detected that I'm using an XU4 and set the Kernel to 4.9 with a unique ID for this board. The options I've used were:

DietPi-Config:

- Performance Options > CPU Throttle Up: 85%
- Performance Options > CPU Governor: I choose 'interactive' but, 'performance' might be the right option
- Advanced Options > Swap File: 1 (Auto size)
- Language and Regional Options > Timezone and keyboard
- Security Options > Change root password and Hostname (reboot required)

Software:

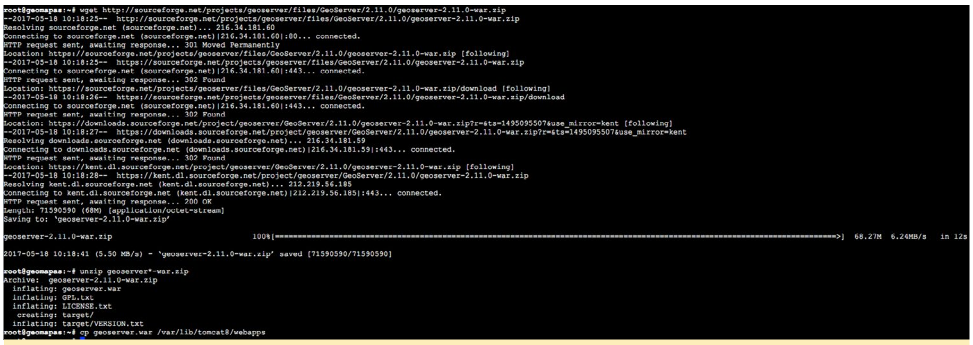
- Fail2Ban
- Tomcat8
- Midnight Commander
- Build-Essentials
- Git Client

You can choose SD/EMMC (not recommended) or an external hard drive device for the user data location. Let everything get installed and grab a coffee while you wait. After a reboot, I checked that SSH server was working and Tomcat was running.

GeoServer

We have two methods to install GeoServer: binary files or a .war file for Tomcat. I chose the second option. You can visit the official home page to install the latest version. At the moment of publication, the latest version is the 2.11.0.

```
$ wget \
http://sourceforge.net/projects/geoserver/\
files/GeoServer/2.11.0/geoserver-2.11.0-war.zip
$ unzip geoserver*-war.zip
$ cp geoserver.war \
/var/lib/tomcat8/webapps
```



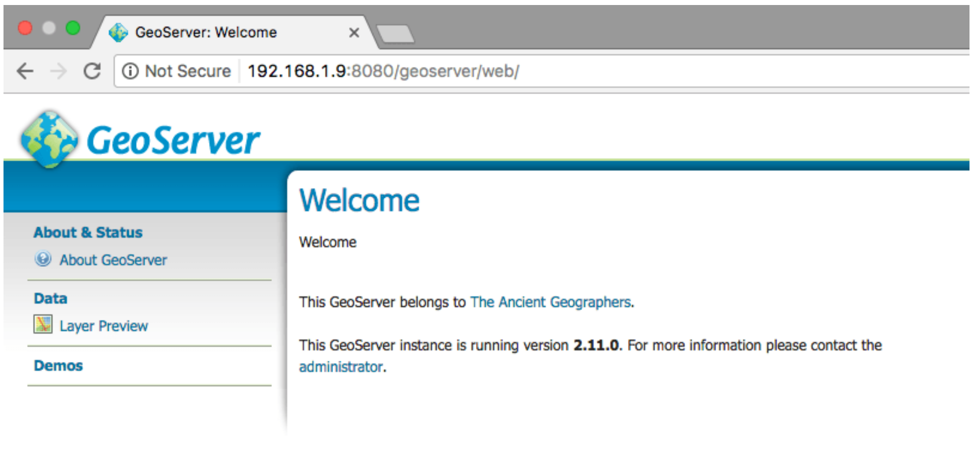
```
root@geomapas:~# wget http://sourceforge.net/projects/geoserver/files/GeoServer/2.11.0/geoserver-2.11.0-war.zip
--2017-05-18 10:18:25--  http://sourceforge.net/projects/geoserver/files/GeoServer/2.11.0/geoserver-2.11.0-war.zip
Resolving sourceforge.net (sourceforge.net)... 216.34.181.60
Connecting to sourceforge.net (sourceforge.net)|216.34.181.60|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://sourceforge.net/projects/geoserver/files/GeoServer/2.11.0/geoserver-2.11.0-war.zip [following]
--2017-05-18 10:18:25--  https://sourceforge.net/projects/geoserver/files/GeoServer/2.11.0/geoserver-2.11.0-war.zip
Connecting to sourceforge.net (sourceforge.net)|216.34.181.60|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://sourceforge.net/projects/geoserver/files/GeoServer/2.11.0/geoserver-2.11.0-war.zip/download [following]
--2017-05-18 10:18:26--  https://sourceforge.net/projects/geoserver/files/GeoServer/2.11.0/geoserver-2.11.0-war.zip/download
Connecting to sourceforge.net (sourceforge.net)|216.34.181.60|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://downloads.sourceforge.net/project/geoserver/GeoServer/2.11.0/geoserver-2.11.0-war.zip?r=&ts=1495095507&use_mirror=kent [following]
--2017-05-18 10:18:27--  https://downloads.sourceforge.net/project/geoserver/GeoServer/2.11.0/geoserver-2.11.0-war.zip?r=&ts=1495095507&use_mirror=kent
Resolving downloads.sourceforge.net (downloads.sourceforge.net)... 216.34.181.59
Connecting to downloads.sourceforge.net (downloads.sourceforge.net)|216.34.181.59|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://kent.dl.sourceforge.net/project/geoserver/GeoServer/2.11.0/geoserver-2.11.0-war.zip [following]
--2017-05-18 10:18:28--  https://kent.dl.sourceforge.net/project/geoserver/GeoServer/2.11.0/geoserver-2.11.0-war.zip
Resolving kent.dl.sourceforge.net (kent.dl.sourceforge.net)... 212.219.56.185
Connecting to kent.dl.sourceforge.net (kent.dl.sourceforge.net)|212.219.56.185|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 71590590 (68M) [application/octet-stream]
Saving to: 'geoserver-2.11.0-war.zip'

geoserver-2.11.0-war.zip                           100%[=================================================================================================>]  68.27M  6.24MB/s   in 12s

2017-05-18 10:18:41 (5.50 MB/s) - 'geoserver-2.11.0-war.zip' saved [71590590/71590590]

root@geomapas:~# unzip geoserver*-war.zip
Archive:  geoserver-2.11.0-war.zip
  inflating: geoserver.war
  inflating: GPL.txt
  inflating: LICENSE.txt
   creating: target/
  inflating: target/VERSION.txt
root@geomapas:~# cp geoserver.war /var/lib/tomcat8/webapps
```

Figure 2 - Setting up GeoServer



Welcome

Welcome

This GeoServer belongs to The Ancient Geographers.

This GeoServer instance is running version **2.11.0**. For more information please contact the administrator.

Figure 3 - GeoServer running

Now the *.war* file is uncompressed in the filesystem. It might take a couple of minutes, after which your map server is ready!

Navigate to the upper right of the web interface to log into GeoServer. The default administrator credentials are as follows:

User name: admin
Password: geoserver

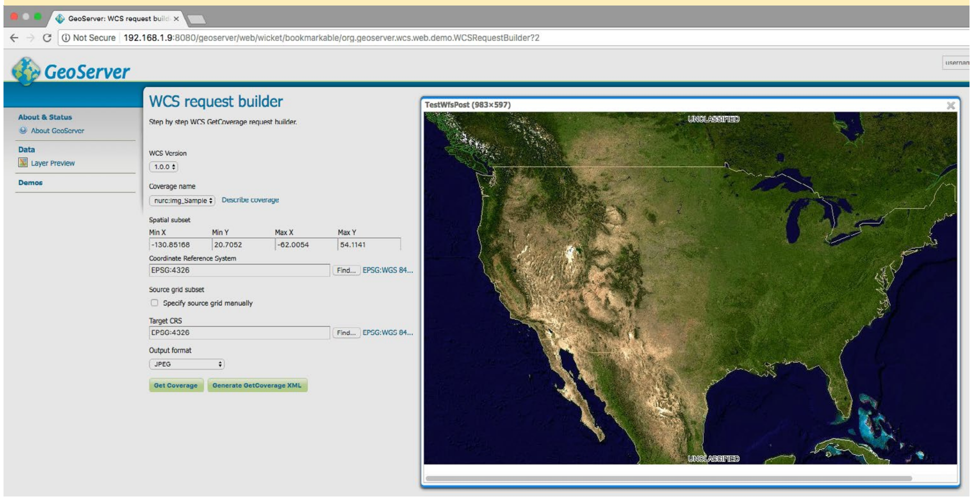
GDAL

GeoServer supports a good range of raster formats, but sometimes non-standard formats and GDAL come to the rescue. We need to install it by typing the following command into a terminal window:

```
$ sudo apt-get install gdal-bin
```

Now you need to install an extension into Tomcat. Go to the Geo Server Download page, click on extension, download the right library and copy it into the /var/lib/tomcat8/webapps/geoserver/WEB-INF/lib/ directory. You have a good partner with gdal tools. The command gdalinfo helps you to get information about raster files. This is very useful, believe me!

Figure 4 - GeoServer with maps loaded



WCS request builder

Step by step WCS GetCoverage request builder.

WCS Version: 1.0.0

Coverage name: nurc:Img_Sample — Describe coverage

Spatial subset

Min X	Min Y	Max X	Max Y
-130.85168	20.7052	-62.0054	54.1141

Coordinate Reference System: EPSG:4326 — EPSG:WGS 84...

Source grid subset

☐ Specify source grid manually

Target CRS: EPSG:4326 — EPSG:WGS 84...

Output format: JPEG

Loading some files

We can select a demo in the left menu to check if it is working, as shown in Figure 4. On the Layer Preview, you can see some examples. Just click on Openlayers to see it in action.

Enhancements

We still have a lot of work to do now on the system side:

- Secure the SSL connections
- Enforce security
- Configure Map Server to save and use files from an external storage
- A map viewer such as Leaflet
- An admin panel using Laravel, since we want an easy to use panel for control of the system and its resources
- Farmer and user panel so that they can see statistics and add or remove layers

Final Notes

There is a lot of documentation to help get a map server running with no effort. Some benchmarks show us that an ODROID-XU4 has the capacity to serve maps and rasters. It might be a bit slower that we want for production, but for now, it's a perfect testing platform.

GEIGER COUNTER

DETECTING RADIATION WITH AN ODROID-C2

by @jojo

In this radiation data logger project, I used an ODROID-C2 and a Geiger tube-based detector board. This initial writeup addresses just the basic hardware setup and minimal programming. In a future article, I would like to present data visualization charts on a website, searchable through dates and times. If anyone would like to contribute towards that effort, you are welcome to contact me.

For now, I have just imported the log files into LibreOffice Calc to create charts manually. Figure 1 shows my setup.

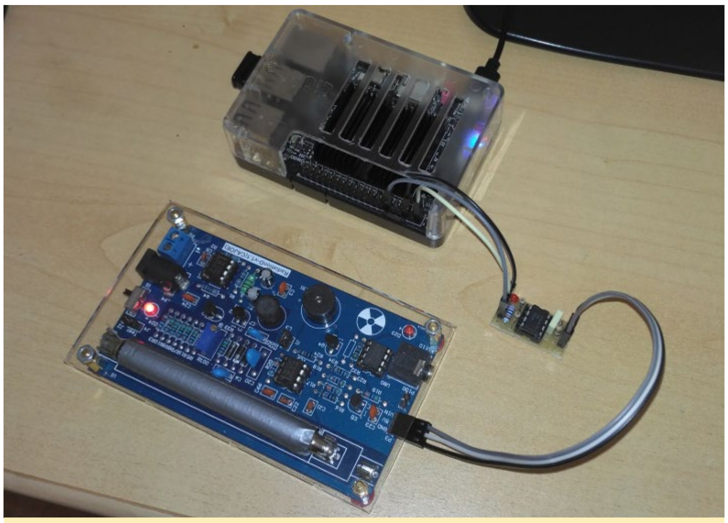


Figure 1 - The Geiger counter attached to the ODROID-C2

Hardware

The inexpensive detector board that I used can be obtained from several online shops like Aliexpress. I faced two problems in using them, however:

1. The Geiger tube (J305 beta) seems to be very light-sensitive. I do not know if this is normal, but it was really strange to see radiation levels periodically rising each day. I observed that it was when the sun was shining onto my desk, so I painted the glass tube black and additionally covered it with non-transparent tape. This reduced the light sensitivity drastically.
2. The "driver stage" on the detector board is extremely weak. The board was not able to drive the input port pin of the C2 to a defined level, so I had to add another driver stage. I used a LMC555 (CMOS variant of the NE555) in mono-stable operation mode. When triggered by the detector board, the LMC555 gives a pulse of 500µs length, which is good to deal with the weak and sometimes "dirty" signal from the detector board.

Figure 2 - Drive board top side

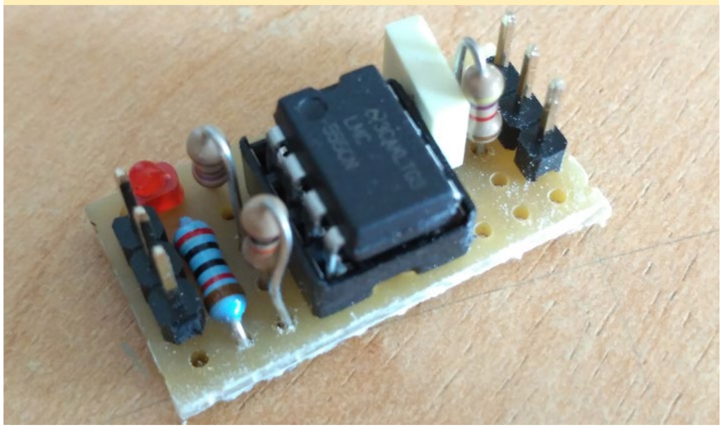
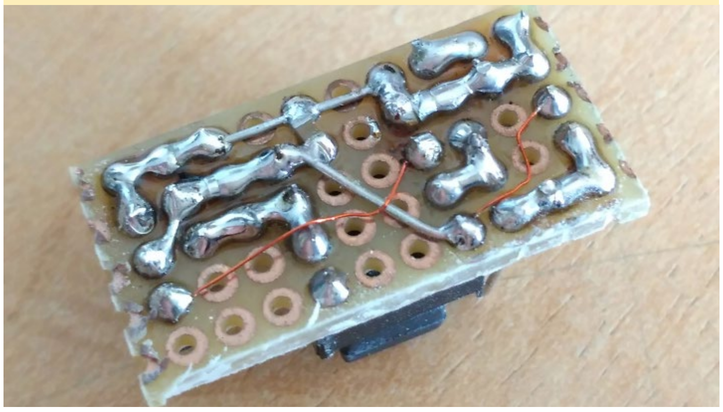


Figure 3 - Driver board bottom side



Both the detector board as well as the driver stage are supplied directly by the GPIO header of the C2, and no additional power source is needed.

Software

I wrote a small C program, which was mainly inspired by this project at http://bit.ly/2tmPv0N. I added the features of data filtering and data updates to the console and a file. The program will create a new file at the moment it is started the first time. If it is running until the beginning of a new day, it will create a new log file each day. The log file's name will include the date and time to make it easy to sort them in a file browser. The program is not pretty and not finished yet, but for now, it works as designed.

You can download the source file (geiger_counter_v6.c) from the original post at http://bit.ly/2s0plwC.

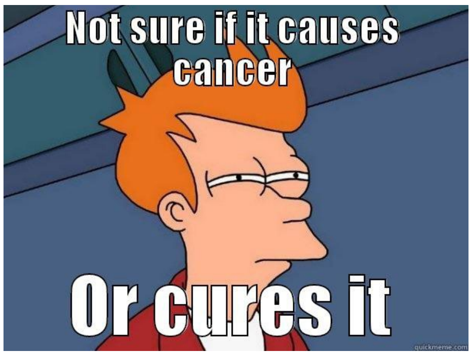
After the download has completed, compile the code:

```
$ gcc -o <your-download-directo-
ry>/\
geiger_counter_v6.c \
-lwiringPi -lpthread
```

Then, simply start the program:

```
$ sudo Documents/Geiger_Counter
```

This project is not "rocket science", and is more of a beginners project. Hopefully it inspires everyone to experiment further.

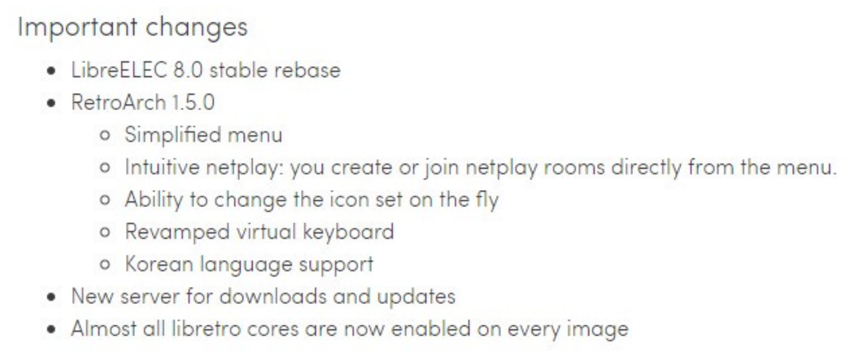


LAKKA 2.0

A GREAT EMULATION OPERATING SYSTEM RECEIVES A MAJOR UPDATE

by @synportack24

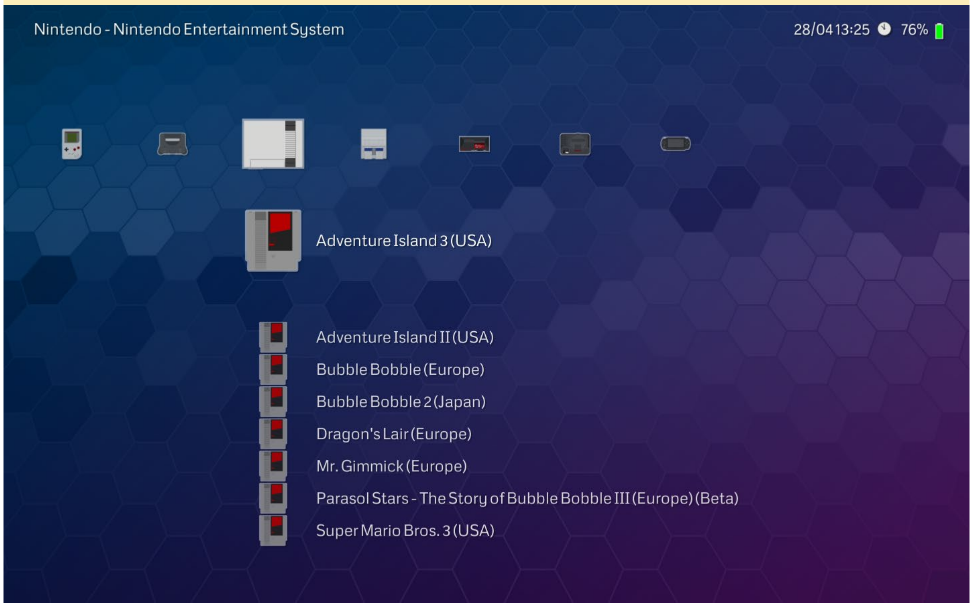
Lakka is an game emulation operating system that first graced the pages of ODROID magazine in 2015 with support for the ODROID-C1. Lakka has come a long way in that time, with a long list of features and cores being added with every release. Now, with version 2.0, Lakka has full support for all Hardkernel ODROID devices. As many avid retro gamers are



Important changes

- LibreELEC 8.0 stable rebase
- RetroArch 1.5.0
  - Simplified menu
  - Intuitive netplay: you create or join netplay rooms directly from the menu.
  - Ability to change the icon set on the fly
  - Revamped virtual keyboard
  - Korean language support
- New server for downloads and updates
- Almost all libretro cores are now enabled on every image

Figure 1 - Playing NES games on Lakka 2.0


Nintendo - Nintendo Entertainment System
28/04 13:25 76%
Adventure Island 3 (USA)
Adventure Island II (USA)
Bubble Bobble (Europe)
Bubble Bobble 2 (Japan)
Dragon's Lair (Europe)
Mr. Gimmick (Europe)
Parasol Stars - The Story of Bubble Bobble III (Europe) (Beta)
Super Mario Bros. 3 (USA)

well aware, there is no shortage of emulation-focused operating systems available for each ODROID platform.

One feature that has always kept me coming back to Lakka was its clean interface and easy-to-use update feature, and Lakka 2.0 does well at keeping that tradition front and center. Right from the start, you will be greeted with the familiar simple linear interface, with the exception of a more animated background. For me, the most welcome feature this update brought was the ability to connect to a wireless network right from the GUI.

I did some light testing of the new version of Lakka on an ODROID-C2 and ODROID-XU4 using a wired XBOX360 controller. All the necessary drivers for my XBOX and many other controllers come pre-installed, so I simply plugged it in and was ready to play. After connecting to my WiFi network, I went to settings and enabled Samba sharing to quickly copy over a couple test games. Neither the XU4 nor the C2 had any problems with any of the SNES or Sega Genesis games. The XU4 understandably had an easier time playing through N64 games. Lakka supports many emulators, or "cores" as they are known, and a single system might have multiple "cores" which you can use to play a game.

What's New

Here are some of the top highlights which are available in Lakka 2.0:

- A new intuitive netplay mode, which means that if you have some friends in a different place and still want to play games with them, you can
- Based on LibreELEC updated to 8 and RetroPi 1.5
- New cores and games were added, including Easy RPG, UAE4ARM (Amiga emulator), VICE (Commodor 64 emulator), PocketCDG (karoke player), and Mr.Boom (bomerman clone)
- Controller support for all major controllers and controller adapters (such as Wii U GameCube Adapter)
- Profile support

Check out Lakka and download a premade image for the ODROID-C2 or ODROID-XU4 at www.lakka.tv.

Figure 2 - Lakka 2.0 supports Netplay rooms

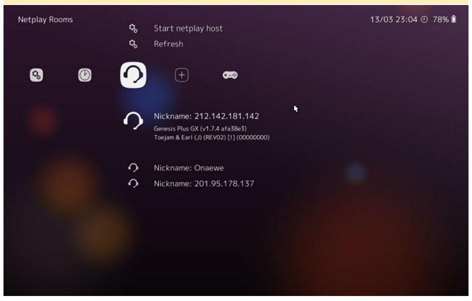
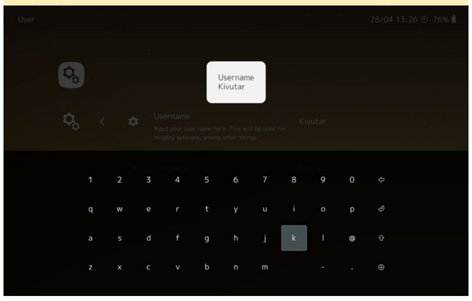
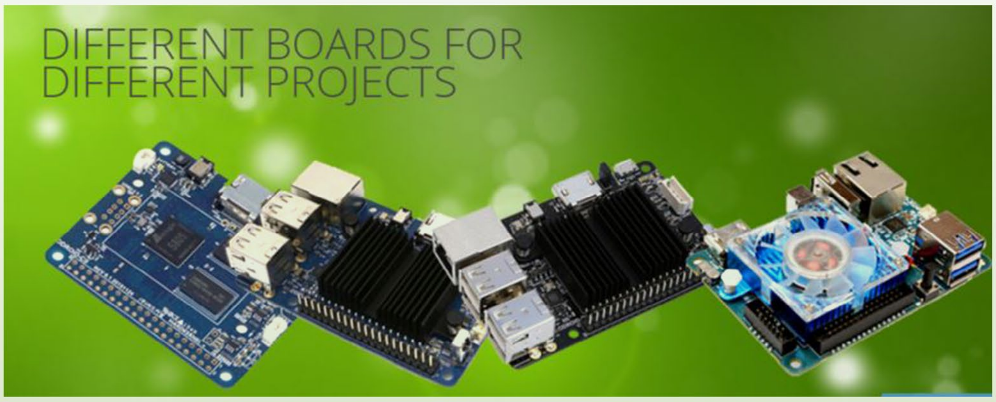

Netplay Rooms 13/03 23:04 78%
Start netplay host
Refresh
Nickname: 212.142.181.142
Genesis Plus GX (v1.7.4 afa38e3)
Toejam & Earl (J) (REV02) [!] (00000000)
Nickname: Onaewe
Nickname: 201.95.178.137

Figure 3 - Creating a username


User 28/04 13:26 76%
Username
Kivutar
Username Kivutar
Input your user name here. This will be used for netplay sessions, among other things.

ODROID



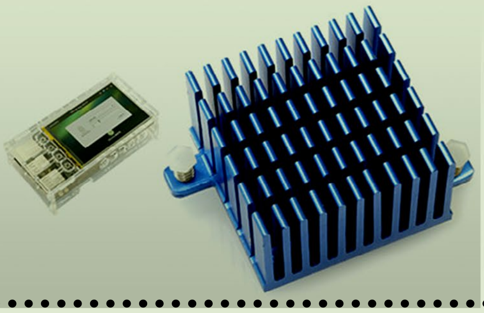
Single Board Computers
Add-On Boards
Cables
Cameras
Cases
Connectivity
Connectors
Coolers
Development
Display
Memory
Power & Battery
Sensors
Sound

Visit the NEW store for all your ODROID needs!

odroidinc.com

MANY KINDS OF ACCESSORIES

Multiple accessory options gives users the ability to create all-in-one, integrated projects such as: tablets, game consoles, infotainment systems, and embedded systems.



MEET AN ODROIDIAN

MICHEL CATUDAL (@MINOU666)

edited by Rob Roy (@robroy)

Please tell us a little about yourself.

I am 66 years old and live in White Pigeon Michigan. I have a wife named Betty, who works at home, and a 32-year old son named Pierre, who is responsible for shipping at ACTIA Corporation.

I was born in Rochebaucourt, Abitibi, a Québec northern town, about 550 miles northeast of Montréal, where it is very cold in the winter. We eventually moved south near Montréal, and I was fascinated by the warmer winters of southern Québec. We were extremely poor, and my parents had 14 kids. Our house had a lot of rats, so my love for cats grew very strong. We later moved to a new house (with no rats) a few miles away after my older sisters, mom and dad found work in the local textile factory.

In 1969, I went to college to become a technologist. Financial help was not very good, so the following year I left college to work in Montréal. After the Canadian military occupied Montréal, I decided to learn the language of the enemy, so I moved to Toronto. I didn't speak much English at the time, so it was complicated to find work, but thanks to a few local French Canadians, I did find some work. We would gather near the Cabbagetown Salvation Army where people would pick up workers every morning for farm work and other types of work. I eventually came back to Québec to finish my technologist degree and then went on to become an engineer.

Every summer from 1969 to 1979, I would work on farms in Ontario. It was often rough due to the hatred that the Anglos had for us. For us, Ontario was a foreign land, and locals were very racist toward us, but farmers loved us because we would to any hard work required, with no questions asked. I could live a whole winter in Québec with my summer income. Today, Mexicans are doing the kind of hard work we were doing when I grew up, and I have great respect for them.

Figure 1 - Michel in his home office with one of his cats, Bella

Figure 2 - Michel's cats are the real bosses of the house

One day, I met my future wife in Missouri. According to her mom, she is part Cherokee, Amish and Irish. I am part Mohawk, Abenakis and French (Québec, Acadie and Louisiane). According to a researcher on my family, I have some English roots from a British slave who married into the family.

I graduated university with an Electronic Technologist and an Electrical Engineer degree from Trois-Rivières, Québec. Work was hard to find, so I accepted a job in Reed City, Michigan. While in Reed City, I did miscellaneous electronic designs from voice recognition to capacitive touch and pinch protection for windows and sunroofs. I worked in Reed city for three years in the early 1980s, and 7 years in the 1990s. I worked in Syracuse from 1985 to 1986 on a mail system for Merrill Lynch for the World Trade center. I also designed a mill counter for Acurite after the company in Syracuse was sold out. I then moved to Buffalo to Cambride Instruments, only to lose that work later when the company got sold to Leica. After that, I went to work in Montréal for a 3-year contract, where I designed the light system controls for the new hockey forum and the casino.

When the contract was over, I briefly worked for Corexco. I was angered by the fact that the pay offer was dramatically cut after I signed up to work for them. I then left and went back to Reed City to work on a new pinch protection design for Ford. When my employer lost court cases against GM, ST and Amway I was let go. I have been working at ACTIA Corp in Indiana ever since.

For the past 15 years, I have been doing embedded software for the automotive, marine and military industry, from intelligent gauges to clusters. Since last year, I do only embedded Linux on Color LCD display boards with a TI DRA726 processor.

Figure 3 - Michel's son Pierre works at the same company as his father

How did you get started with computers?

I built my first computer (Cosmac) in 1976. I worked later on with a ZCPR computer and IBM PC. ZCPR was a hacker's version of CPM/80é.

What attracted you to the ODROID platform?

I did not know anything about ODROIDs until someone asked me to help out with getting Lazarus to work with hardfloat on the ODROID-U2.

How do you use your ODROIDs?

My interest in ODROIDs has been strictly the challenge to do something that very few people can do and to get more familiar with embedded Linux. I am designing some boards for the ODROID-C2, such as one with a CAN device from Microchip. Unlike the Beaglebone, it doesn't have an on-board CAN.

Which ODROID is your favorite and why?

My favorite is the XU3, which is both powerful and fast. Even without full GPU support, the speed and power of the miscellaneous cores makes up for it. I will probably get an XU4 once the rush for my work at ACTIA is over.

What innovations would you like to see in future Hardkernel products?

I would like to see better support for GPU on Linux, and a board with a device that has CAN.

What hobbies and interests do you have apart from computers?

Linux computers are my hobby. It is only recently that I do embedded Linux as part of my work. My embedded Linux work is a chance for me to do something dramatically different from my work on Windows at ACTIA Corp.

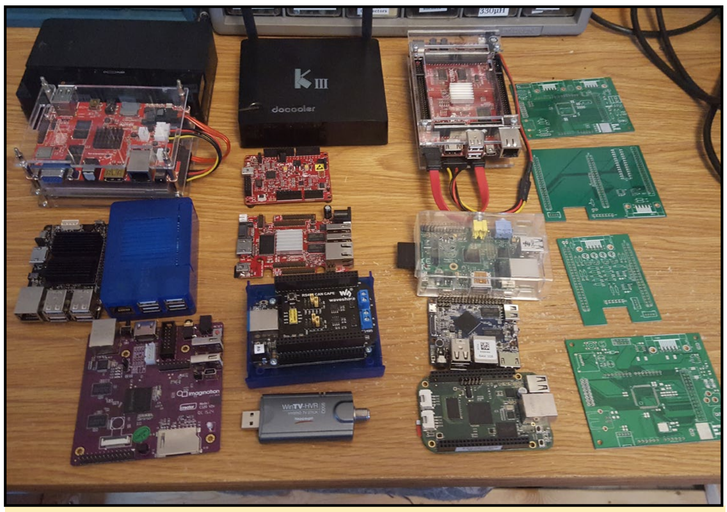


Figure 4 - Michel has an impressive collection of single board computers

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

Don't be afraid of the challenges!