

# ODROID

Magazine

Año Tres  
Num. #36  
Dic 2016

# ODROID

## *Refrigerado por agua*



Crea una configuración moderna para conseguir el máximo y absoluto rendimiento de tu ordenador

- Usar las funciones del codificador de Hardware del ODROID-XU4

- Cómo usar un ODROID como altavoz Bluetooth



# Qué defendemos...

Nos esmeramos en presentar una tecnología punta, futura, joven, técnica y para la sociedad de hoy.

Nuestra filosofía se basa en los desarrolladores. Continuamente nos esforzamos por mantener estrechas relaciones con éstos en todo el mundo.

Por eso, siempre podrás confiar en la calidad y experiencia que representa la marca distintiva de nuestros productos.

Simple, moderno y único.

De modo que tienes a tu alcance lo mejor para hacer todo lo que imagines



## HARDKERNEL



Realizamos envíos de ODROID-C2 and ODROID-XU4 a los países de la UE! Ven y visita nuestra tienda online!

Dirección: Max-Pollin-Straße 1  
85104 Pförring Alemania

### Teléfono & Fax

telf : +49 (0) 8403 / 920-920

email : service@pollin.de

Nuestros productos ODROID se pueden encontrar en: <http://bit.ly/1tXPXwe>

The screenshot shows a web browser displaying the Pollin Electronic website. The search bar at the top contains 'Suche - Pollin Electronic'. Below it, a search result for 'odroid' is shown with 18 results. The results include various ODROID components like the ODROID-U3 BACKUP BATTERY, ODROID-U3 CASE, and ODROID-U3 eMMC Modul. Each item has a small image, price, and a brief description. The website has a blue and white color scheme with a navigation bar at the top.



**H**ardkernel asistió a la TechCon 2016 a finales de octubre, y mostró algunas de las posibilidades del nuevo **ODROIDC2**. Había varias demos de proyectos de bricolaje, incluyendo una pantalla Ambilight, un servidor CloudShell 2 que ejecuta el kernel más reciente y un reproductor multimedia con pantalla táctil que usa un Shield HiFi. La configuración de refrigeración líquida que aparece en la portada de este mes es otro ejemplo de increíble proyecto de bricolaje que permite crear el sistema definitivo para sacar el máximo rendimiento al overclock. Es una obra de arte funcional que demuestra la experiencia técnica de la comunidad **ODROID**.

Otros proyectos que han creado los miembros de la comunidad incluyen el uso de un **ODROID** como un altavoz Bluetooth, el desarrollo de un kernel en tiempo real para su uso en un sistema operativo de un único hilo de ejecución y el diseño de una combinación seedbox y una unidad de almacenamiento en red desde un **ODROID-XU4** utilizando el **CloudShell**. Andy concluye su tutorial de Docker con una introducción al modo swarm, nuestro científico loco Bo detalla sus últimos descubrimientos, Tobias nos ayuda a seleccionar los mejores mandos para jugar con una análisis en profundidad, y Bruno nos trae lo último en diversión **Android** con sus juegos favoritos de este mes.

ODROID Magazine, que se publica mensualmente en <http://magazine.odroid.com/>, es la fuente de todas las cosas ODROIDianas. • Hard Kernel, Ltd. • 704 Anyang K-Center, Gwanyang, Dongan, Anyang, Gyeonggi, South Korea, 431-815 • fabricantes de la familia ODROID de placas de desarrollo quad-core y la primera arquitectura ARM "big.LITTLE" del mundo basada en una única placa.  
Para información sobre cómo enviar artículos, contacta con [odroidmagazine@gmail.com](mailto:odroidmagazine@gmail.com), o visita <http://bit.ly/lyplmXs>.  
Únete a la comunidad ODROID con miembros en más de 135 países en <http://forum.odroid.com/> y explora las nuevas tecnologías que te ofrece Hardkernel en <http://www.hardkernel.com/>



**HARDKERNEL**



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**

# NUESTRO MARAVILLOSO PRESONAL ODROIDIAN:



## Rob Roy, Editor Jefe

Soy un programador informático que vive y trabaja en San Francisco, CA, en el diseño y desarrollo de aplicaciones web para clientes locales sobre mi cluster ODROID. Mis principales lenguajes son jQuery, angular JS y HTML5/CSS3. También desarrollo SO precompilados, Kernels personalizados y aplicaciones optimizadas para ODROID basadas en las versiones oficiales de Hardkernel, por los cuales he ganado varios Premios. Utilizo mis ODROIDs para diversos fines, como centro multimedia, servidor web, desarrollo de aplicaciones, estación de trabajo y como plataforma de juegos. Puedes echar un vistazo a mi colección de 100 GB de software ODROID, kernel precompilados e imágenes en <http://bit.ly/1fsaXQs>.



## Bruno Doiche, Editor Artístico Senior

Algunos se preguntan por qué Bruno habla consigo mismo. Él respondió: "Por supuesto que hablo conmigo mismo. ¡A veces necesito asesoramiento experto, hombre! "Sí, él ha hecho muchas cosas locas en la revista durante años. Después de todo, actuar con normalidad nunca ha sido una de sus mayores fortalezas.



## Manuel Adamuz, Editor Español

Tengo 31 años y vivo en Sevilla, España, aunque nací en Granada. Estoy casado con una mujer maravillosa y tengo un hijo. Hace unos años trabajé como técnico informático y programador, pero mi trabajo actual está relacionado con la gestión de calidad y las tecnologías de la información: ISO 9001, ISO 27001, ISO 20000. Soy un apasionado de la informática, especialmente de los microordenadores como el ODROID, Raspberry Pi, etc. Me encanta experimentar con estos equipos y traducir ODROID Magazine. Mi esposa dice que estoy loco porque sólo pienso en ODROID. Mi otra afición es la bicicleta de montaña, a veces participo en competiciones semiprofesionales.



## Nicole Scott, Editor Artística

Soy una experta en Producción Transmedia y Estrategia Digital especializada en la optimización online y estrategias de marketing, administración de medios sociales y producción multimedia impresa, web, vídeo y cine. Gestione múltiples cuentas con agencias y productores de cine, desde Analytics y Adwords a la edición de vídeo y maquetación DVD. Tengo un ODROID-U3 que utilizo para ejecutar un servidor web sandbox. Vivo en el área de la Bahía de California, y disfruta haciendo senderismo, acampada y tocando música. Visita mi web <http://www.nicolecscott.com>.



## James LeFevour, Editor Artístico

Soy un especialista en medios digitales que disfruta trabajando como freelance en marketing de redes sociales y administración de sitios web. Cuanto más aprendo sobre las posibilidades de ODROID más me ilusiona probar cosas nuevas con él. Me trasladé a San Diego desde el Medio Oeste de los EE.UU. Continuo muy enamorado de muchos de los aspectos que la mayoría de la gente de la Costa Oeste ya da por sentado. Vivo con mi encantadora esposa y nuestro adorable conejo mascota; el cual mantiene mis libros y material informático en constante peligro.



## Andrew Ruggeri, Editor Adjunto

Soy un ingeniero de sistemas Biomédicos anclado en Nueva Inglaterra que actualmente trabaja en la industria aeroespacial. Un microcontrolador 68HC11 de 8 bits y el código ensamblador son todo lo que me interesa de los sistemas embebidos. Hoy en día, la mayoría de los proyectos en los que trabajo están en lenguajes C y C++, o en lenguajes de alto nivel como C# y Java. Para muchos proyectos, utilizo placas ODROID, pero aún sigo intentando utilizar los controladores de 8 bits cada vez que puedo (soy un fan de ATMEL). Aparte de la electrónica, soy un amante de la fotografía analógica y desarrollo la película friki con la que disfruto intentando hablar en idiomas extranjeros.



## Venkat Bommakanti, Editor Adjunto

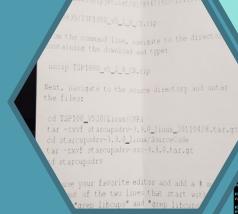
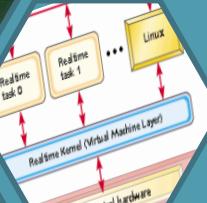
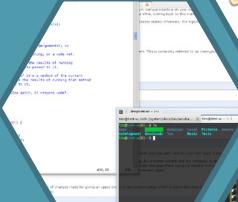
Soy un apasionado de los ordenadores desde la bahía de San Francisco en California. Procuro incorporar muchos de mis intereses en proyectos con ordenadores de placa reducida, tales como pequeñas modificaciones de hardware, carpintería, reutilización de materiales, desarrollo de software y creación de grabaciones musicales de aficionados. Me encanta aprender continuamente cosas nuevas, y trato de compartir mi alegría y entusiasmo con la comunidad.



## Josh Sherman, Editor Adjunto

Soy de la zona de Nueva York, y ofrezco mi tiempo como escritor y editor para ODROID Magazine. Suelo experimentar con los ordenadores de todas las formas y tamaños: haciendo trizas las tablets, convirtiendo Raspberry Pi en PlayStations y experimentando con los ODROIDs y otros SoCs. Me encanta trabajar con los elementos básicos y así poder aprender más, y disfrutar enseñando a otros escribiendo historias y guías sobre Linux, ARM y otros proyectos experimentales divertidos.

# INDICE

	HARDKERNEL EN LA ARMCON - 6
	JUEGOS CON EXAGEAR - 8
	MANDOS PARA JUEGOS - 10
	JUEGOS ANDROID: MISSILES! - 14
	IMPRESORA TERMICA DE TICKETS - 15
	BOOT.INI PERSISTENTE - 16
	HIFI DOMINACION DEL MUNDO - 17
	FUNCIONES DEL CODIFICADOR DE HARDWARE - 18
	ODROID COMO UN ALTAVOZ BLUETOOTH - 23
	KERNEL EN TIEMPO REAL - 24
	SYNERGY - 25
	ODROID CON REFRIGERACION LIQUIDA - 26
	DOCKER - 29
	SEEDBOX - 34
	CONOCIENDO UN ODROIDIAN - 41

# HARDKERNEL EN LA ARM TECHCON 2016

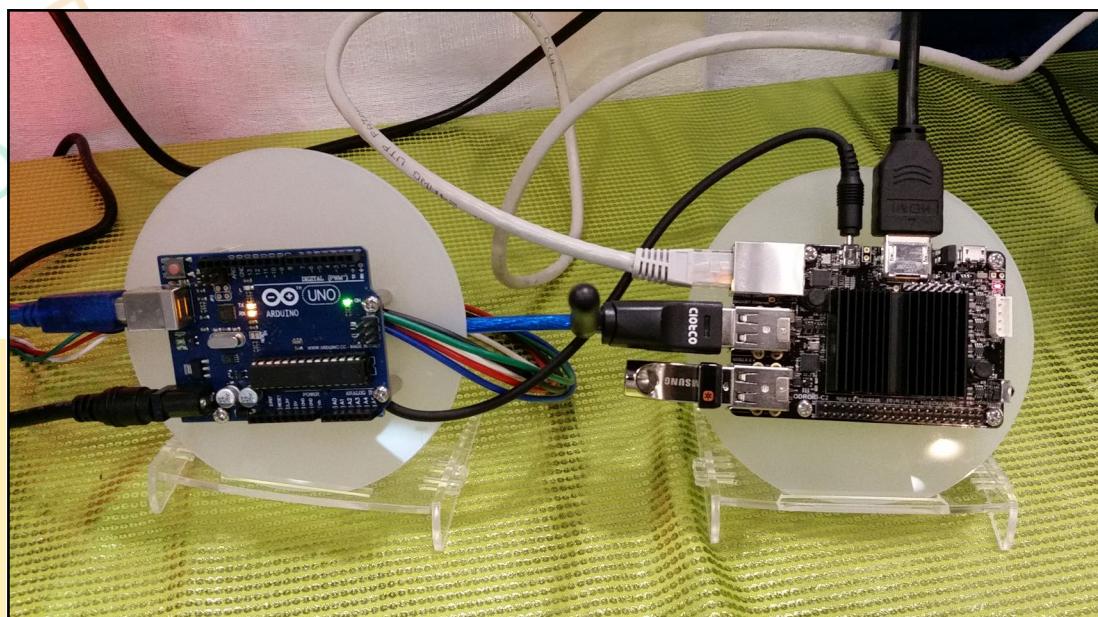
## PRESENTANDO EL ODROID-C2

por Rob Roy (@robroy)

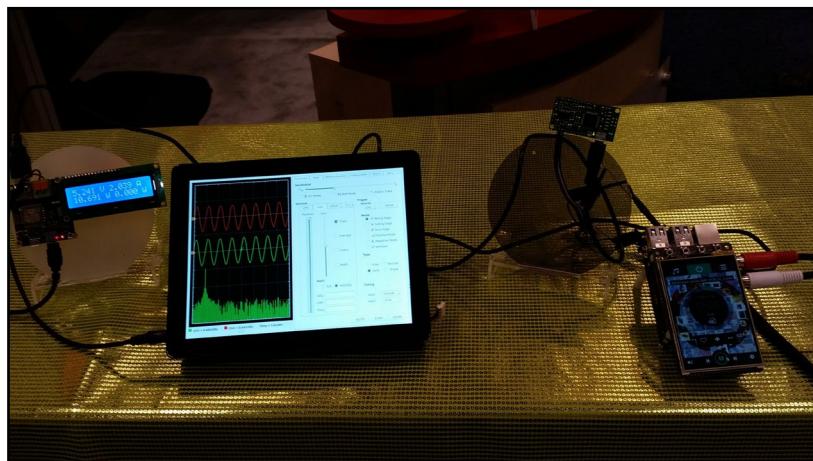
Este año en la ARM TechCon en Santa Clara, California, los ingenieros de Hardkernel presentaron varios proyectos utilizando el ODROID-C2, incluyendo una configuración Shield HiFi con un reproductor portátil Volumio y una magnífica pantalla 4K Ambilight de 55 pulgadas. Muchos ODROIDians se pararon a admirar algunos de los productos de la próxima generación, como la nueva pantalla táctil ODROID-VU8 de 8 pulgadas con carcasa y el nuevo CloudShell 2, configurado para ejecutar una matriz RAID utilizando dos discos duros SATA. ¡Echa un vistazo a las fotos para ver los productos que Hardkernel ofrecerá muy pronto!



El stand de Hardkernel con una pantalla Ambilight llamaba mucho la atención



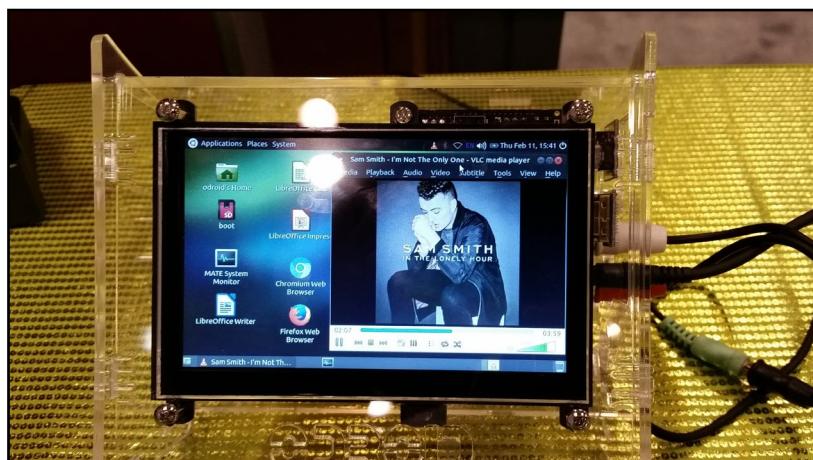
El sistema Ambilight se ejecutaba sobre un Arduino que analizaba el vídeo 4K en tiempo real mientras que el ODROID-C2 utilizaba Kodi para reproducir el vídeo simultáneamente en el monitor 4K de 55 pulgadas



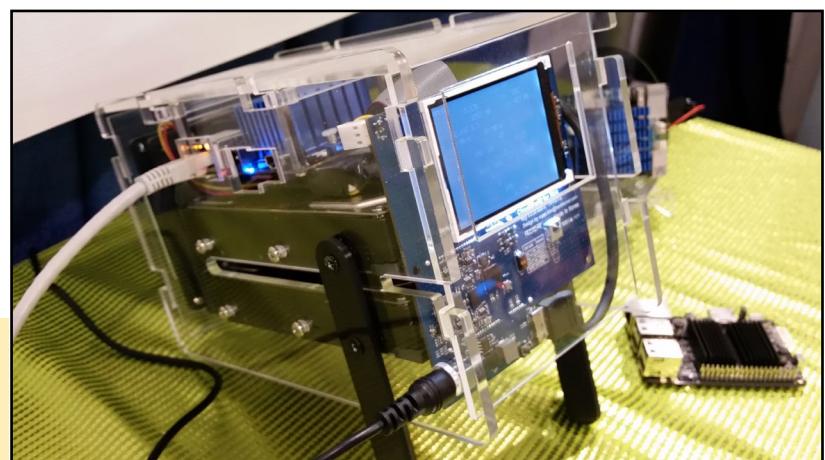
El ODROID-C2 de la derecha reproduce música a través del Shield HiFi y el ODROID-C2 de la izquierda está ejecutando el software osciloscopio en la nueva pantalla táctil ODROID-VU8 de 8 pulgadas con carcasa.



Música de alta calidad reproducida a través de un Shield HiFi durante la conferencia



El más excepcional de todos los dispositivos:  
Una tablet con pantalla táctil y Ubuntu 16.04



El nuevo CloudShell 2, que alberga un ODROID-XU4 ejecutando la versión 4.7.8 del kernel Linux y una matriz RAID SATA

# JUEGOS RETRO CON EXAGEAR

por Gaukhar Kambarbaeva

No es ningún secreto que los juegos retro han vuelto a estar de moda. Tras comprar un nuevo gadget, cualquier jugador quiere al momento descargar y jugar a sus títulos favoritos. Por suerte para los poseedores de un ODROID, existe una forma de convertir su dispositivo en una máquina de juegos retro con Exagear Desktop y así poder disfrutar de sus juegos favoritos de Linux y Windows. En este artículo, voy a describir cómo ejecutar algunos juegos de PC sobre la plataforma ODROID: Arcanum, Heroes of Might y Magic 3 y Sid Meier's Alpha Centauri.

## Instalación

Antes de empezar, instala Exagear Desktop desde <http://bit.ly/2cul90r>. Coloca el archivo ExaGear Desktop, los paquetes de instalación y la clave de licencia de ExaGear Desktop en la misma carpeta. Abre tu terminal MATE (línea de comandos), desplázate a esta carpeta y desempaquetta el archivo:

```
$ tar -xvzpf exagear-desktop-
odrxu4.tar.gz
```

El siguiente paso es instalar y activar ExaGear en nuestro ODROID. Puedes hacerlo con el script de instalación de Exagear dentro de la carpeta, automáticamente se detectarán los paquetes y la clave de licencia:

```
$ sudo ./install-exagear.sh
```

La instalación es relativamente fácil y debería completarse sin problemas.

## Empecemos

ExaGear Desktop básicamente ejecuta una versión virtual de Ubuntu que usa la arquitectura x86, en lugar de la arquitectura ARM que utiliza nuestros ODROIDS. Vamos a iniciar el entorno desde nuestro Terminal MATE y echamos un vistazo:

```
$ exagear
```

Puedes confirmar que te encuentras en un entorno x86 ejecutando el comando "arch":

```
$ arch
i686
```

Recomendamos que actualices tus repositorios apt-get durante el primer inicio del sistema invitado:

```
$ sudo apt-get update
```

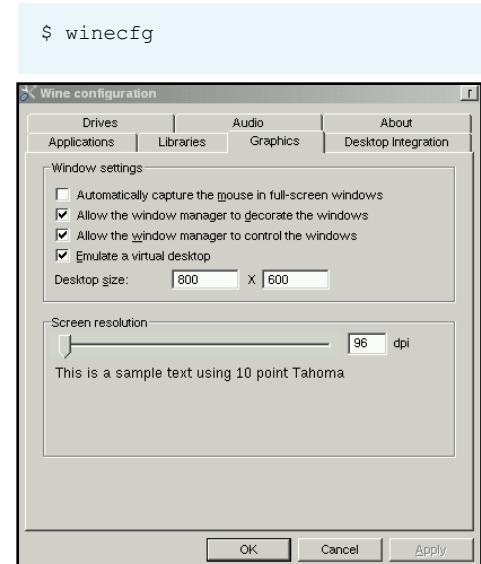
Puesto que vamos a ejecutar juegos de Windows en este entorno, también necesitamos instalar Wine. Wine existe para Ubuntu de forma nativa en nuestros ODROIDS, pero todavía no existen juegos ARM para Windows. Necesitamos instalar Wine x86 dentro del entorno virtual de Exagear. Esto se puede hacer fácilmente usando el co-



mando apt-get:

```
$ sudo apt-get install wine
```

Después de instalar Wine, recomendamos ejecutar winecfg y activar la barra "Emulate a virtual desktop". De lo contrario, te puedes encontrar con problemas cuando intentes cambiar al modo de pantalla completa las aplicaciones:



**Figura I - Pantalla de configuración de Wine**

Llegados a este punto, estamos preparados y listos para intentar instalar algún juego de PC. Es posible ejecutar todo tipo de distribuciones e instaladores, pero el proceso es bastante complicado y puede llevarte algún tiempo averiguar cómo funciona cada juego, especialmente juegos antiguos que tienen

problemas de compatibilidad. Recomendamos utilizar versiones de juegos sin DRM desde servicios como Good Old Games (GOG), que ofrece la posibilidad de descargar e instalar juegos que ExaGear Desktop normalmente admiten.

## Arcanum

Este impresionante juego de aventuras del 2000 puede ejecutarse en ExaGear Desktop. Primero, compra y descarga Arcanum (<http://bit.ly/2fjmVa0>). A continuación, ejecuta el instalador con Wine en el terminal MATE y aparecerá la interfaz gráfica del instalador del juego:

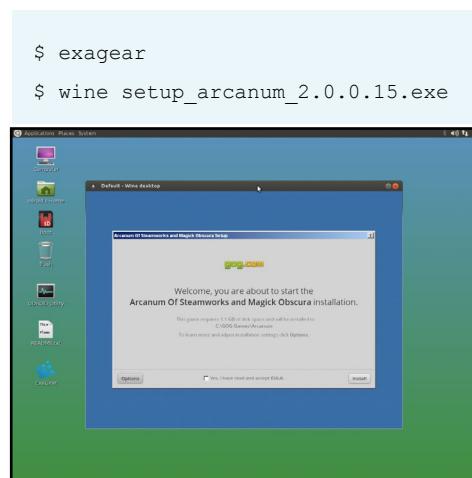


Figura 2 - Arcanum

Una vez finalizada la instalación, puedes ejecutar Arcanum desde tu escritorio. Si ExaGear instala un ícono en el escritorio principal de tu ODROID con Ubuntu, debería iniciarse automáticamente el entorno virtual y cargar el juego. ¡Selecciona tu héroe favorito y empieza a jugar a Arcanum en tu ODROID!

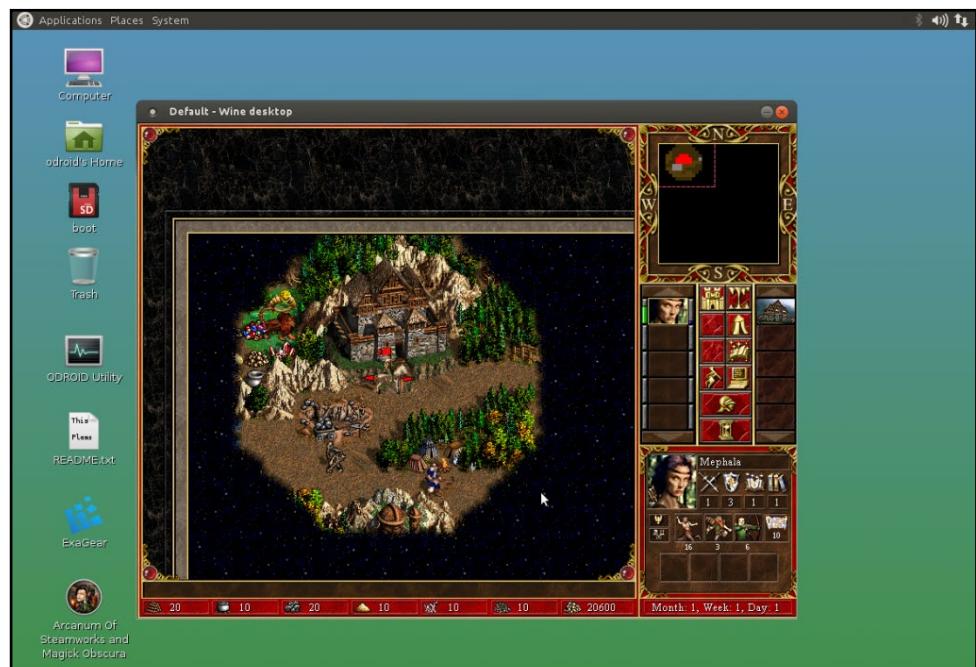


Figura 4 - Heroes of Might y Magic 3 en acción

## Heroes of Might and Magic 3

Al igual que Arcanum, puedes conseguir HoMM3 desde GOG en <http://bit.ly/2gftTtl>. Este juego de estrategia por turnos también se carga normalmente con el instalador y Wine en un terminal MATE:

```
$ exagear
$ wine setup_homm3_complete_2.0.0.16.exe
```

Una vez cargado, aparecerá el instalador del juego y podrás continuar con la instalación. Una vez finalizada la instalación, puedes ejecutar HoMM3 directamente desde el escritorio. Espera a que se cargue el juego, luego verás el menú Inicio, desde el que podrás seleccionar el mapa y jugar a Heroes of Might and Magic 3 en tu ODROID.

## Sid Meier's Alpha Centauri

Este impresionante juego de estrategia de los creadores de Civilization también está disponible en GOG en <http://bit.ly/2fyVBf>. Puesto que utilizas el instalador GOG, puedes descargarlo e instalarlo en el sistema X86 invitado con

Wine, al igual que los otros juegos:

```
$ exagear
$ wine setup_sid_meiers_alpha_centauri_2.0.2.23.exe
```



Figura 5 – Captura de pantalla de Alpha Centauri

Una vez finalizada la instalación, puedes ejecutar Alpha Centauri directamente desde el escritorio o desde el menú Inicio. Selecciona una facción e inicia tu aventura de colonización espacial.

Exagear Desktop (<http://bit.ly/2cul90r>) te permite ejecutar una gran cantidad de juegos de PC de los 90 y principios del 2000 en tu ODROID. También puedes ejecutar otras aplicaciones x86. Para comentarios, preguntas y sugerencias, visita el post original en <http://bit.ly/2fH6Fwf>.

# JUEGOS LINUX

## ELIGE EL MANDO CORRECTO SEGUN TU ESTILO DE JUEGO

por Tobias Schaaf (@meveric)



Normalmente suelo hablar de los diferentes juegos y emuladores que funcionan en los ODROIDS, pero en este caso me gustaría hablar de uno de los activos más importantes para los jugadores: el gamepad. Hay un montón de gamepads por ahí, quisiera hablar de unos cuantos que poseo, con qué juegos los utilizo y cuales prefiero utilizar. Intentaré explicar cómo configurar tu propio mando basándonos en mi imagen de GameStation Turbo de ODROID, que también se puede aplicar a mis imágenes de Debian Jessie. También me detendré en algunos casos especiales.

### **Trustmaster Dual Trigger 3 en 1 Rumble Force**

Tengo algunos mandos que he ido recopilando a largo de los años, algunos de los cuales los quería/necesitaba para crear imágenes compatibles. El primero que recibí fue de una tienda alemana, hace casi 15 años, mientras que todavía

vivía con mis padres. Hasta ese momento sólo usaba joysticks, ese fue mi primer gamepad.

Tenía este mando (<http://bit.ly/2fo53WY>) para mi PC por aquel entonces, sabiendo bien que como jugador de PC, rara vez se utiliza un mando. Mientras que la mayoría de los juegos se ejecutan con un ratón y teclado, algunos funcionan con joysticks, especialmente las simulaciones espaciales y de combate. Sin embargo, la capacidad de respuesta era a menudo insuficiente. Por aquel entonces culpaba a Windows, porque cada vez que calibraba el joystick, éste funcionaba bien durante un tiempo, pero luego se desconfiguraba. Hoy día, sé que el stick analógico izquierdo no funciona correctamente y siempre apunta hacia la dirección superior izquierda, lo cual hace que sea casi imposible usar este gamepad.

A parte de eso, el mando realmente es muy bueno. Tiene un tacto muy agradable ya que usa algún tipo de acabado en goma. Esto le proporciona un agarre muy bueno y evita que se deslice con los dedos. Es un poco más pesado que otros mandos, pero no excesivamente. Además, viene con 12 botones, un D-Pad, dos sticks analógicos y dos disparadores en la parte posterior. Tiene soporte para vibración y dos conectores uno para PS2 y otro USB para PS3 y PC. También cuenta con 4 botones en el borde superior, los cuales se agradecen a la hora de trazar mapas. El gamepad es programable, pero nunca me he tomado la molestia de programarlo.

**Figura 1 – Gamepad Trustmaster Dual Trigger 3 en 1 Rumble Force, fue mi primer Gamepad**



#### Pros

- Muy buen tacto y sensibilidad, se apoya muy bien sobre la mano
- Botones muy buenos y robustos, si zarandeas el gamepad nada hace ruido, se percibe que los botones son de alta calidad.
- Cuatro botones en la parte superior + 2 botones de disparo (botones adicionales para funciones especiales)
- Soporte y símbolos de PlayStation (bueno para la emulación PSP o PS1)

#### Contras

- Modelo muy antiguo, probablemente difícil de encontrar
- No todos los emuladores detectan o utilizan soporte para vibración
- Tiene algunos problemas con Retroarch
- Sin soporte para la mayoría de los emuladores

### **Hama Black Force**

El gamepad de Hama "Black Force" es un gamepad que conseguí mucho más tarde. Es un modelo más económico, aunque sigue siendo un gamepad bastante bueno. Es una copia barata de un mando de PS3, pero funciona. Ninguno de los botones tiene una descripción que no sea números. Hay un botón "analógico" en el lugar donde está el botón de inicio en el mando PS3. Si pulsa el botón, se iluminará un LED rojo que está por debajo del botón que indica que el modo analógico está activado. Realmente es



**Figura 2 – Gamepad Hama Black Force, por cortesía de game-debate.com**

necesario para enviar la señal analógica del D-Pad, o sino la mayoría de los emuladores y programas no reconocerán el D-Pad para nada. El mando tiene 4 botones en la parte superior, aunque le faltan los disparadores.

Este gamepad es detectado como un “joystick USB genérico”, aunque funciona con Retroarch y otros programas y emuladores. El mando es de menor calidad y si lo zarandeas, se puede oír como los botones hacen ruido. La mayor parte del ruido parece provenir de los botones de la parte superior, pero también en algunos casos del D-Pad. Los sticks analógicos tienen una superficie áspera que aportan un agarre extra, que no es malo, pero no es tan cómodo como el acabado de goma del mando real de PS3. Han creado un nuevo modelo con las marcas de los botones X, O, Cuadrado y Triángulo de la Playstation.

#### Pros

- Barato - este gamepad está pensado para aquellos que cuentan con un bajo presupuesto, los clones están normalmente entre los 5 y 15 euros
- Puede parecer que no se trata de un buen mando, pero hay muchas críticas positivas de este mando. Por lo general es “bueno” pero no es nada del otro mundo.
- Funciona con la configuración de fábrica con muchos juegos, programas y emuladores
- Fácil de instalar y configurar para la mayoría de los emuladores y juegos

#### Contras

- No es de una “marca comercial” para que pueda ser reconocido por algún programa. De modo que otros mandos “sin marca” con diferentes esquema de botones pueden ser reconocidos con el mismo nombre, lo cual hace que sea casi imposible tener diferentes archivos de configuración para estos mandos
- El cable de 1,9 m puede ser demasiado corto para algunos jugadores.

## Mando inalámbrico Xbox 360

Cuando empecé a trabajar en la ODROID GameStation Turbo, sabía que necesitaba probarla con uno de los “principales” mandos que existen. Resultó que los mandos de la Xbox 360 eran los más ampliamente compatibles, por lo que tenía sentido hacerme con uno. Además quería tener un mando inalámbrico para deshacerme del cable. Puesto que estaba desarrollando un sistema dirigido a los niños, no quería que éstos tropezaran con los cables.

Este mando es probablemente el más pesado de todos los que tengo, pero su tacto es muy bueno sobre las manos y pienso que su peso no será demasiado problema. Es el principal mando con el que mi imagen OGST está configurada, todos los juegos deberían funcionar por defecto con este mando. Hay algunos

**Figura 3 - El mando inalámbrico de la Xbox 360 es probablemente el mejor mando compatible con Linux**



problemas menores con los drivers: el mando siempre parpadea, o puede no ser reconocido tras apagarlo y volviendo a encender cuando estás dentro de un juego o un programa como Kodi. Aparte de eso, probablemente sea el gamepad que mejor soporte Linux. Esto no significa que no vaya a recibir parches y correcciones de errores con el tiempo, pero sí que la mayoría de los juegos/programas están desarrollados con soporte para este mando.

Algunos juegos carecen de todo menos de soporte para el mando de la Xbox 360. La variante inalámbrica necesita un adaptador de PC remoto de un tercero, ya que no utiliza un estándar como Bluetooth, pero esto realmente no es un inconveniente, ya que el adaptador soporta hasta 4 mandos simultáneamente. Esto significa que con sólo usando un puerto USB, podemos utilizar hasta 4 gamepads independientes. También viene con pilas AA estándar, o puedes comprar un pack de pilas recargables y usar un cargador para tener tus mandos siempre a punto. Siempre tendrás un mando operativo, siempre y cuando tenga pilas en casa o que tengas pilas cargadas.

Lo que no me gusta de este mando es la llamada “zona muerta” y la incapacidad para volver a “0” (cero) cuando sueltas los controles. Los sticks analógicos van de -32768 a +32768 de izquierda a derecha o de arriba a abajo. Cuando los sueltas debería volver a 0. La escala en la que tiende a alejarse de 0 es a la que nos referimos como “zona muerta”. Significa que cualquier valor en esa escala se debe ignorar y considerarse 0 para evitar movimientos no deseados. En los buenos mandos estas zonas muertas suelen ser muy pequeñas. Por ejemplo, es muy común decir que una zona muerta está en el 4-5% del valor máximo posible, que sería 1300-1700. Incluso si vas un poco más allá y dices que se ignoren los valores por debajo de 2000, debería funcionar con la mayoría de los mandos

Con los mandos Xbox 360, cuando soltaba el stick analógico para que

volviera a 0, observaba valores muy altos, en torno a +/- 7500 o superior. Esa es una zona muerta de más del 20%. Los dos mandos anteriores tienen valores entre 0 y 255, de modo que el centro debe estar en torno a 128. Normalmente no se desvían más del 124 o 132 que es alrededor de un 3-4% y la mayoría incluso estaban más cerca. El mando de la PS3 tiene valores entre -127 y +127 y siempre vuelve a 0 cuando se suelta. Nunca observe que se diera el caso de que no volviese a 0.

#### Pros

- Mando mejor compatible con juegos, emuladores y programas
- Hasta 4 mandos simultáneamente a través de un puerto USB
- Facilidad para cambiar las pilas, por lo que siempre está listo para jugar
- No es demasiado caro y es fácil de comprar, hay un montón de diferentes modelos y diseños para elegir

#### Contras

- No es muy preciso
- Algo pesado en comparación con otros mandos

## Mando DualShock 3 de Playstation 3

El mando de la Playstation 3, a menudo llamado "Sixaxis", utiliza comunicaciones Bluetooth estándar. Es el mando preferido para dispositivos móviles como tablet y smartphones, aunque a menudo estos necesitan ser "rooteados". Se pueden utilizar tanto con un cable USB como con un adaptador Bluetooth en el ODROID. Esto hace que sean muy

**Figura 4 - Mando Sixaxis DualShock 3 de Playstation 3**



**Figura 5 – Inclínate cuando gires en tu juego de carreras favorito**

flexibles, aunque sólo se puede conectar un mando PS3 por dongle Bluetooth.

Este mando es sin duda uno de los mejores que existe. Tiene un tacto muy bueno, es resistente, relativamente fácil de usar, tiene un buen soporte y resulta entretenido jugar con él. Si por casualidad conectas la PS3 a un PC Linux o Windows que pueda leer todas las entradas desde el gamepad, podrás comprobar la infinidad de funciones que tiene este mando. Presionar un botón no es simplemente 0 y 1, sino que en realidad registra lo fuerte que presionas el botón.

El mando PS3 tiene un sensor de movimiento o giroscopio que registra cómo se gira el mando. Cuando se trata de funciones, el mando PS3 es probablemente el mejor que tengo. También es el más caro. Una nuevo puede costar entre 50 y 90 €.

#### Pros

- Muy buenos controles, muy resistente, buena sensación en las manos
- Control muy preciso
- Muchas funciones
- Con opción inalámbrica y por cable.

#### Contras

- Muy caro
- Se carga a través del USB (sin baterías intercambiables)

## Mando de juegos Sabrent USB 2.0 con 12 botones para PC

Este es el último gamepad que compré. La razón fue que quería un gamepad



**Figura 6 – Mando Sabrent con 12 botones barato y funcional**

con 6 botones para emuladores como yabause para jugar a juegos de la Sega Saturno. Se necesitan 6 botones en lugar de los 4 que son los que trae la mayoría de los mandos de hoy en día. También debería funcionar bien con algunos juegos MAME que necesitan 6 botones en lugar de 4.

Este mando se registra como joystick USB genérico, igual que el mando Hama, lo que me lleva de nuevo al hecho de que no es posible disponer de una configuración para diferentes mandos con este tipo de mandos. También cuenta con 12 botones, al igual que el gamepad de Hama, pero simplemente están ordenados de un modo diferente.

En el centro hay tres botones: "Mode", que es el mismo que el botón "Analog" en el gamepad de Hama y es necesario para que el D-pad funcione, y los botones "Turbo" y "Clear" que no parecen ser reconocidos como botones adicionales, pero deben tener algún significado interno. Esto significa que faltan dos botones que normalmente se utilizarían como "Start" y "Select". Estos botones aparecen como botones 5 y 6 en el mando. Los 4 botones de la parte superior son una vez más sólo botones y no tiene disparadores. Normalmente uso L2 y R2 como sustitutos de "Start" y "Select". El D-pad también es diferente. En realidad, este es el único mando que tiene botones completamente independientes para el Dpad. En todos los demás mandos, el D-pad es de un modo u otro

una cruz y todas las teclas de dirección están conectadas, pero no en el Sabrent.

También he probado la zona muerta en este mando, y mientras que el stick analógico derecho está perfectamente centrado en 128 y se mantiene así (los valores están entre 0 y 255), el stick analógico izquierdo no vuelve al centro y salta constantemente entre 135 y 140 en lugar de 128, lo que significa que se desvía alrededor de 5-10%. Puesto que este mando actúa de un modo similar al mando Hama, también funcionará en Retroarch sin tener que configurar nada.

#### Pros

- Barato y ligero
- Cable muy largo 3m (aproximadamente 9 pies)
- El diseño de 6 botones es muy bueno para la Sega Saturn y otros emuladores
- Funciona de serie con retroarch

#### Contras

- De mala calidad y poco duradero
- Para mi gusto, este mando es demasiado pequeño. Debería ser un poco más grande, con los botones un poco más separados. Además se vuelve un poco incómodo en la mano después de un rato
- Los botones Turbo y Clear deberían funcionar como Start y Select
- Un stick analógico no está centrado, pero este no es el único mando que tiene este problema. La zona muerta no es tan mala.

Ya he comentado que estos mandos normalmente son detectados automáticamente por Retroarch y otros emuladores. Sin embargo, ¿funcionan todos de serie sin tener que configurarlos? La respuesta es lamentablemente, no, pero eso no significa que no puedas usarlos.

En mi imagen ODROID GameStation Turbo (OGST), tengo muchos emuladores y juegos que utilizan mandos, así que vamos a conocer cuáles son los

que funcionan. Por ahora, tengo cuatro emuladores que merecen la pena ejecutar con mandos: Retroarch - para la mayoría de los emuladores, PPSSPP - emulador de Playstation Portable, reicast - emulador de Sega Dreamcast, y Yabause - un nuevo emulador de Sega Saturn.

Retroarch es en realidad el mejor emulador cuando hablamos de soporte para gamepad y joystick. En mi imagen está configurado para que utilice udev y así detectar el mando, cuenta con un gran número de archivos de configuración para diferentes mandos que permite que éstos funcionen directamente sin tener que realizar configuraciones. Son compatibles alrededor de unos 130 mandos diferentes, hay una muy alta probabilidad de que tu mando sea compatible.

Los mandos de la XBox 360 y PS3 funcionan 100% con Retroarch. También lo hace el mando de Hama, y para la mayoría de los juegos, probablemente el Sabrent también. El Trustmaster se detecta, pero debido a su defecto en el stick analógico izquierdo, te puede causar algunos problemas. No todos los mandos se detectan correctamente cuando hablamos de soporte para vibración. El emulador N64 y algunos otros núcleos pueden usar vibración pero, por alguna razón, no siempre funciona.

PPSSPP realmente no tiene buen soporte para mandos. Oficialmente sólo cuenta con un mapeo para los mandos de XBox 360, pero que en realidad no funciona en los ODROIDs, así que creé el mío. También he añadido soporte para el mando PS3, que debería funcionar directamente sin tener que configurar nada, pero no estoy seguro. Puedes configurar el mando PS3 manualmente, gracias a mis cambios y asignar correctamente todos los botones que necesites.

Ninguno de los otros mandos son oficialmente compatibles, por lo que su uso puede no estar garantizado. Por lo general, todos los mandos deben funcionar, ya que PPSSPP utiliza SDL2 como backend, lo cual debería permitir un cierto nivel de compatibilidad. Siem-

pre y cuando asigneas todas las teclas que necesites, debería funcionar bien con cualquier mando.

Reicast es algo complicado. Oficialmente también tiene únicamente soporte para el mando XBox 360, pero yo he añadido archivos de configuración para PS3. Los otros mandos no son compatibles y es poco probable que funcione directamente. No es posible reasignar los botones dentro del emulador como en PPSSPP pero hay una forma que podría funcionar, que detallaré en algún momento más adelante.

Desde la última actualización, Reicast soporta vibración. Durante mis pruebas, todos, excepto los mandos Trustmaster y PS3, funcionaban con Reicast y soportaban la vibración. El Trustmaster no tiene soporte para vibración y aunque el mando PS3 se detecta con vibración, era tan débil que apenas se podía sentir o simplemente no funcionaba.

Con Yabause, tiene la interfaz GTK o Qt, ambas configuradas por separado. Los botones se pueden configurar desde el emulador. Bajo mi experiencia, una vez que el D-pad y los botones están configurados, éstas parecen funcionar con cada mando. Puesto que puedes reconfigurar los botones desde el emulador, esto no supone un gran problema.

Como se ha comentado anteriormente, tener una configuración para diferentes "Joysticks Genericos USB" puede no funcionar, ya que no todos están configurados del mismo modo. También observé que el mando PS3 en el C2 sólo funciona a través de la conexión inalámbrica Bluetooth, la conexión por cable no funciona.

## Idoneidad

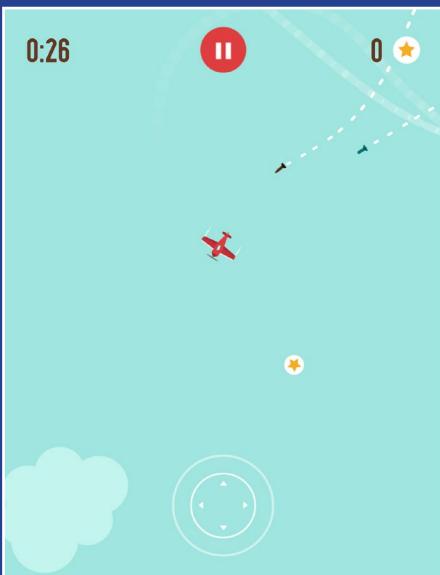
Normalmente tengo toda mi imagen configurada para usar los mandos de XBox 360. Esto facilita las cosas, ya que todo tiene la misma configuración y no tienes que cambiar nada. Recientemente he creado mi propia configuración para poder cambiar entre mandos dependiendo de los juegos que ejecute.

# MISSILES!

## NO HAY MEJOR MANERA DE VOLAR QUE ESQUIVAR ATAQUES INCESANTEMENTE

por Bruno Doiche

El cielo está azul y te encuentras volando pacíficamente en tu querido avión... Entonces de repente empiezas a temer por tu vida. Aparecen misiles por todas partes. Has sido entrenado para sobrevivir más de 10 segundos y la dificultad sigue subiendo. Aunque es extremadamente difícil, este juego te mantendrá pegado a tu pantalla!



Tan pronto como te familiarices con el juego, activa al modo rápido. ¡Tendrás el doble de la diversión en la mitad de tiempo!

<https://play.google.com/store/apps/details?id=pl.macaque.Missiles>

Aún juego a todos los núcleos Libretro en Retroarch usando un mando XBox 360. A menudo es el mando más adecuado, además puedo controlar Kodi con él muy bien. Puedo navegar por los menús sin ratón y teclado, lo cual resulta muy agradable, sobre todo porque el mando XBox 360 es inalámbrico. Puesto que Retroarch utiliza una configuración automática para configurar los mandos, puedes cambiar fácilmente de un mando a otro si lo deseas. También uso el mando XBox 360 para Dreamcast, ya que funciona muy bien con él.

Para PPSSPP, cambié al mando PS3. Es mucho más adecuado usar un mando de Playstation para una consola de Playstation, puesto que todos los botones coinciden y además es muy sensible.

Me hice con el Sabrent para jugar a los juegos de Sega Saturn en el emulador de Yabause, pero realmente no uso el Hama. Lo usaba en el pasado cuando aún no tenía mi mando Xbox 360, pero últimamente sólo lo uso en un ODROID de repuesto por si necesito probar algo y no quiero cambiar mi adaptador remoto para el mando XBox 360.

No uso el Trustmaster para nada, porque el stick analógico izquierdo no funciona bien, a menudo causa problemas. En realidad me decepcionó bastante, porque es un mando muy bueno que tiene un buen tacto en las manos.

### Cómo configurar tu mando para OGST

Si tiene un mando que no es compatible con los emuladores que he mencionado, aquí tienes una pequeña guía para poder configurarlo:

Retroarch: debería haber una herramienta instalada llamada "retroarch-joyconfig". Con esa herramienta, puede crear un archivo de configuración para tu mando. Ésta generará un resultado en formato texto, que debe copiar y pegar en un nuevo archivo en /usr/share/libretro/autoconfig/udev/ que utiliza privilegios de root. Una vez creado, tu mando debería ser detectado automáticamente.

PPSSPP: Ve al menú Settings -> Controls para configurar tu mando y asignar los botones.

Yabause: Al igual que PPSSPP, puede configurar los controles desde el propio emulador. Por ejemplo, presiona CTRL + S en yabause-qt.

Reicast: Para Reicast, se utiliza una herramienta llamada reicast-joyconfig que es similar a retroarch-joyconfig. Está escrita en Python y requiere un módulo python llamado evdev, que puede que tengas que instalar a través de pip. Después, es similar a Retroarch a la hora de crear el archivo de configuración. Este archivo debe copiarse en /usr/local/share/reicast/mappings usando privilegios de root. Luego, debes ajustar el archivo de configuración de Reicast en /home/odroid/.reicast/emu.cfg siguiendo la guía del foro <http://bit.ly/2ggdO9Y>. También es necesario crear una entrada para el archivo de asignaciones similar al siguiente para poder cargar la nueva configuración de tu mando:

```
evdev_mapping_1= mycontroller.cfg
```

También tengo una herramienta instalada llamada antimicro que te permite crear configuraciones únicas para tus mandos. Puede asignar las teclas del teclado a tu mando en caso de que un juego o un emulador no te permitan utilizar un mando. Incluso puede asignar los movimientos del ratón y los botones de éste a un mando y utilizar el mando como sustituto.

### Reflexiones finales

Cualquiera que sea el mando que utilices, posiblemente haya una manera de hacerlo funcionar en mis imágenes pre-compiladas de ODROID. Algunos usuarios incluso han sido capaces de conectar auténticos joysticks arcade que utilizan al completo. Cualquiera que sea el mando que elijas, debería haber una forma de hacer que tu experiencia de juego sea agradable en los ODROIDS y mis imágenes OGST.

# INSTALANDO IMPRESORA DE TICKETS TERMICA EN LINUX

## CRONICAS DE UN CIENTIFICO LOCO

por Bo Lechnowsky (@respectech)

“Algún día, todo será sin papeles”, murmuras en voz baja al mirar tu espacio de trabajo repleto de componentes electrónicos, cables y papeles. En el futuro, sólo tendrá que abrir paso a través de la electrónica y los cables para encontrar ese módulo eMMC perdido. Pero ahora mismo, buscas este módulo eMMC o tarjeta microSD perdida en tu espacio de trabajo es como buscar una aguja en un pajar, donde la mayoría de “heno” está hecho de papel!

Observas que muchos de los papeles son impresiones útiles que contienen información importante. “La mayoría de estas notas son bastante pequeñas, pero abarcan una hoja entera de papel!” Empiezas a pensar en el problema del tamaño del papel. Entonces recuerdas haber visto una impresora térmica abandonada desde hace algún tiempo procedente del anterior proyecto de dominación del mundo, depositada dentro de una caja en una esquina de tu laboratorio subterráneo. La coges y ves que es una impresora térmica STAR TSP100 con una interfaz USB.

“¡Eso es todo!” Empiezas a pensar en cómo puedes usar una impresora térmica para imprimir notas de cualquier tamaño. Muchas de las notas serán tan pequeñas que podrías grabarlas en los objetos físicos sobre los que estás haciendo las notas. “Puedo añadir una impresora térmica a la consola de mi flota de vehículos en caso de necesitar imprimir cualquier cosa que me surja en el camino!”

Tu primer intento es simplemente añadirla vía “cups”, accesible a través del menú de Linux “System/Administra-

tion/Printers” o “System Tools/Printers”. Asignar el driver de impresora “Generic Text Only” no funcionó y no hay ninguna descarga para Linux/ARM en el sitio web del fabricante.

Entonces, ¿Cómo podemos imprimir con una impresora térmica desde Linux con un ODROID? Investigas e imprimes los siguientes pasos (sobre una hoja de papel de tamaño normal por ahora) para tener una referencia:

En una ventana terminal, introduce:

```
$ sudo apt update
$ sudo apt install libcurl2-dev
$ sudo apt install libcurlimage2
$ sudo apt install libcurlimage2-dev
```

A continuación, descarga un paquete que contiene el código fuente de Linux para la impresora STAR TSP100 desde <http://bit.ly/2fPybtO>. Desde la línea de comandos, navega hasta el directorio que contiene el archivo zip de descarga y escribe:

```
$ unzip TSP100_U_v5_2_0_CD.zip
```

Después, ve al directorio de las fuentes y descomprime los archivos:

```
$ cd TSP100_V520/Linux/CUPS
$ tar -zxf starcupsdrv-3.3.0_
linux_20110428.tar.gz
$ cd starcupsdrv-3.3.0_linux/
SourceCode
$ tar -zxf starcupsdrv-src-
3.3.0.tar.gz
$ cd starcupsdrv
```

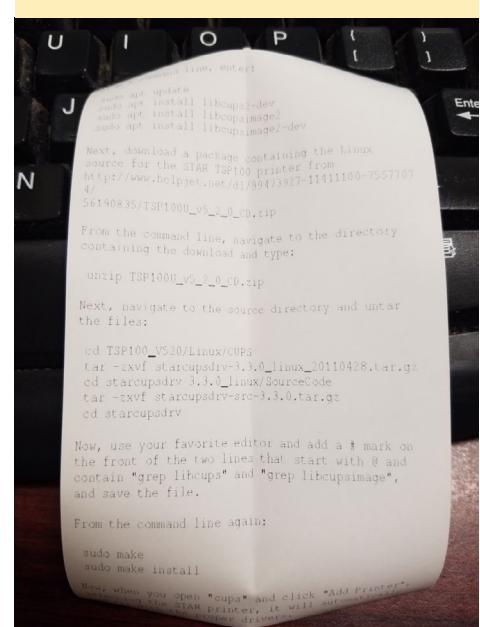
Usa tu editor favorito y agrega una marca # al principio de las dos líneas que empiezan por @ y que contienen “grep libcurl” y “grep libcurlimage”, guarda el archivo. A continuación, escribe lo siguiente en la línea de comandos:

```
$ sudo make
$ sudo make install
```

Ahora, cuando abras “cups” y hagas clic en “Add Printer”, seleccionando la impresora STAR, se instalarán automáticamente los drivers adecuados.

Al reemplazar todas las grandes impresiones de tu espacio de trabajo por impresiones térmicas eficientes en espacio, incluyendo este conjunto de instrucciones, piensas: “¡Hoy he dominado los problemas de organización de mi espacio de trabajo, mañana conquistaré el mundo!”

### Impresora Térmica con ticket impreso desde un ODROID



# BOOT.INI PERSISTENTE

## CONSERVANDO LOS CAMBIOS DURANTE UNA ACTUALIZACION

por Adrian Popa (@mad\_ady)

Imagínate el siguiente escenario: llegas a casa, tarde por la noche, acompañado por una hermosa mujer y quiere impresionarla viendo una película o escuchando música en tu elegante configuración ODROID. Enciendes el ODROID y el televisor y observas que aparece el mensaje “No hay señal”. El led azul parpadea como debería, pero sientes un sudor frío cuando tu cita te pregunta qué es lo que ocurre. Educadamente te excusas y vas al baño un segundo, para “refrescarte”. Utilizas tu teléfono para conectarte vía SSH a tu ODROID para investigar el problema. Pronto te da cuenta de que boot.ini ha sido sobrescrito por una actualización y que la resolución no es compatible con el televisor. Después de juguetear con VI desde tu teléfono y reiniciar el ODROID, regresas a la sala de estar. Ahora la habitación se encuentra iluminada con el resplandor azulado de Kodi y te sientes aliviado. Sin embargo, tu cita se ha percatado de que algo iba mal, y se excusa por la noche. ¿Te preguntas si esto se podría haber evitado?

El problema es que cuando el paquete bootini se actualiza, sobrescribe el archivo /media/boot/boot.ini eliminando así las modificaciones que hubieras hecho en él, como la configuración de la resolución, la activación del soporte DAC o incluso la elección de un sistema de archivos raíz diferente. Los nuevos usuarios suelen tropezar con este problema y hasta ahora, tenía

**Figura 1 - Todas los parámetros del archivo boot.ini están comentados por defecto**

```
#general
#This configuration is parsed by the bootini-persistence script
#uncomment the section you want to override in boot.ini and set the desired value

#####
# C2 configuration #
#####

#C2 resolution
#m=1080p60hz

#C2 BPP Mode
#m_bpp=32

#C2 HDMI/DVI/VGA mode
#vout=dvi

#C2 HDMI HotPlug Detection control
#hpdp=true

#C2 Console
#condev=console=ttyS0,115200n8 console=tty0

#C2 Meson timer
#mesontimer=1

#C2 nographics
#nographics=0

#C2 monitor output
#monitor_onoff=false

#C2 maxcpus
#maxcpus=4

#C2 Max frequency
#max_freq=1752

#C2 bootargs
bootargs=root=UUID=e139ce78-9841-40fe-8823-96a304a09859 rootwait ro ${condev} no_console_suspend hdmmode=${m} ${comde} m_bpp=${m_bpp} vout=${vout} fsck.repair=yes net.ifnames=0 elevator=noop disablehpd=${hpdp} max_freq=${max_freq} maxcpus=${maxcpus} monitor_onoff=${monitor_onoff}
```

```
[general]
#This configuration is parsed by the bootini-persistence script
#uncomment the section you want to override in boot.ini and set the desired value

#####
# C2 configuration #
#####

#C2 resolution
#m=1680x1050p60hz

#C2 BPP Mode
#m_bpp=32

#C2 HDMI/DVI/VGA mode
#vout=dvi

#C2 HDMI HotPlug Detection control
#hpdp=true

#C2 Console
#condev=console=ttyS0,115200n8 console=tty0

#C2 Meson timer
#mesontimer=1

#C2 nographics
#nographics=0

#C2 monitor output
#monitor_onoff=false

#C2 maxcpus
#maxcpus=4

#C2 Max frequency
#max_freq=1752

#C2 bootargs
bootargs=root=UUID=e139ce78-9841-40fe-8823-96a304a09859 rootwait ro ${condev} no_console_suspend hdmmode=${m} ${comde} m_bpp=${m_bpp} vout=${vout} fsck.repair=yes net.ifnames=0 elevator=noop disablehpd=${hpdp} max_freq=${max_freq} maxcpus=${maxcpus} monitor_onoff=${monitor_onoff}
```

**Figura 2 - Se han configurado las opciones deseadas en boot.ini**

que renunciar a las actualizaciones o aprender a vivir con ellas. Renunciar a las actualizaciones es una algo terrible, ya que se está pasando por alto correcciones de errores y nuevas características, como el soporte para el overclock. Para solucionar esto, he realizado algunos cambios en el paquete bootini para restaurar la configuración del usuario tras actualizar un nuevo archivo boot.ini.

Para usar este método, necesitarás editar un archivo llamado boot.ini.default que se encuentra en /media/boot, que es la partición vfat que se utiliza en el proceso de arranque. Este archivo contiene versiones comentadas de todos los valores predeterminados de la configuración de boot.ini que puedes modificar. Este archivo no se sobrescribirá con las actualizaciones, aunque se volverá a crear si lo eliminas.

Tendrás que descomentar las líneas que te interesan y ajustar los valores que quieras para esas variables. Por ejemplo, para el modelo ODROID-C2, puedes configurar cosas como la resolución (“m”), modo de salida de vídeo (“vout”), fre-

**Figura 3 – Resultado en consola del script bootini-persistence.pl**

```
lurian@y-scu1:~$ sudo /usr/share/bootini/bootini-persistence.pl
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: Applying user preferences for boot.ini
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: boot.ini has 156 lines
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: Setting m from 1080p60hz to 1680x1050p60hz
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: vout enabled and set to dvi
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: Setting max_freq from 1536 to 1752
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: Setting monitor_onoff from 0 to 1
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: Setting bootargs from root=UUID=e139ce78-9841-40fe-8823-96a304a09859 rootwait ro ${condev} no_console_suspend hdmmode=${m} ${comde} m_bpp=${m_bpp} vout=${vout} fsck.repair=yes net.ifnames=0 elevator=noop disablehpd=${hpdp} max_freq=${max_freq} maxcpus=${maxcpus} monitor_onoff=${monitor_onoff} to root=UUID=e139ce78-9841-40fe-8823-96a304a09859 rootwait ro ${condev} no_console_suspend hdmmode=${m} ${comde} m_bpp=${m_bpp} vout=${vout} fsck.repair=yes net.ifnames=0 elevator=noop disablehpd=${hpdp} max_freq=${max_freq} maxcpus=${maxcpus} monitor_onoff=${monitor_onoff}
[14-Nov-17 15:54:44] /usr/share/bootini/bootini-persistence.pl: Writing boot.ini finished. Written 159 lines
lurian@y-scu1:~$
```

cuencia máxima (“max\_freq”) así como modificar los argumentos de arranque, como la partición raíz UUID. La Figura 1 muestra el archivo por defecto, mientras que la Figura 2 muestra una versión personalizada. Deja comentados los parámetros que no necesites, los cuales serán ignorados.

Para aplicar las nuevas configuraciones y cambios a boot.ini.default, puedes reconfigurar el paquete bootini con el siguiente comando:

```
$ sudo dpkg-reconfigure bootini
```

Otra posibilidad es simplemente ejecutar el script bootini-persistence.pl:

```
$ sudo /usr/share/bootini/bootini-persistence.pl
```

El resultado que se muestre te indicará qué parámetros se han cambiado, tal y como aparece en la Figura 3.

Para volver al archivo boot.ini original, puede volver a copiar el archivo /usr/share/bootini/boot.ini con el siguiente comando:

```
$ sudo cp /usr/share/bootini/boot.ini /media/boot/boot.ini
```

He enviado una solicitud de GitHub a los ingenieros de Hardkernel para que incluyan estas mejoras en su versión oficial, la cual debería estar disponible pronto. Espero que estos cambios te faciliten las cosas con Linux y con tus ODROIDS presentes y futuros.



**NEVER GIVE UP**

¡La persistencia vale la pena!

# SOFTWARE EQUALIZADOR PARA ANDROID

## CRONICAS DE UN CIENTIFICO LOCO

por bo Lechnowsky (@respectech)

Tu mente corre mientras que te sientes en tu oscuro laboratorio. La última barrera con las que te has encontrado amenaza con desbaratar tus planes de dominación del mundo. “¿Qué ha cambiado?” te preguntas, mientras valoras cada una de las ideas que inundan tu mente. Piensas en los acontecimientos que han conducido a esta desviación inoportuna de tus planes.

Te encuentras conduciendo uno de tus vehículos para recoger algunos suministros para tu última invención. Pusiste en marcha tu unidad de pantalla táctil del coche impulsada por un ODROID-C2 y empezaste a escuchar tu lista de reproducción de “dominación del mundo”. Entonces, te golpea como un némesis, Dr. Usual, prueba su arma de rayo de náuseas sobre ti. Te sientes un poco un poco indispuesto. Tenías la configuración del ecualizador en Android para enfatizar los bajos en tus canciones favoritas pero, siempre que aparecían graves, las voces y otras notas se cortaban y los golpes de graves no tenían ningún efecto sobre ellas.



Tu increíble Software Equalizador en Android.

¿Cómo puedes continuar con tus planes de dominación del mundo sin tener buenos bajos y graves? ¡Es impensable! Después de pasar toda la noche intentando olvidarlo y trabajar en otros inventos, hasta que de pronto ¡Eureka! Estoy seguro de que tiene algo que ver con el nivel de volumen en Android, y la configuración del ecualizador que afecta al volumen cuando se intenta modificar el flujo del audio.

Te diriges a toda prisa al garaje, inicias el sistema del vehículo y compruebas la configuración de volumen en Android. ¡Estaba fijado al 100%, tal y como habías sospechado! Activas rápidamente el sistema de sonido y los subwoofers y aumentas el nivel de volumen en el sistema de sonido mientras reduces el volumen principal a un 50% en Android. Algo nervioso inicias tu lista de reproducción. Te golpea como el arma de rayos de náuseas del Dr. Usual, aunque con mucha más fuerza. “¡Lo hice!” Gritas, con tu voz ahogada por el sonido industrial de la electrónica que golpea los altavoces. “Esto merece una celebración”, piensas, mientras conduces lentamente hacia la heladería, hasta que te das cuenta que son las 5:30 am y aún no está abierta.

# USANDO LAS FUNCIONES DEL CODIFICADOR DE HARDWARE DEL ODROID-XU4

por Marian Mihailescu (@memeka)

**L**os codificadores de hardware utilizan un algoritmo diseñado para codificar vídeo y datos dentro del contenido multimedia, y generalmente es la forma más eficiente de visualizar los videos. Tenemos dos opciones para lograr la codificación por hardware en el ODROID-XU4:

- un FFmpeg personalizado que soporte codificación por hardware, o
- the GStreamer framework available for creating streaming media applications

Los pasos para activar la codificación por hardware que se presentan a continuación se agrupan en pasos comunes y pasos específicos. Ten en cuenta que este tutorial está dirigido a usuarios medios/avanzados y si las cosas salen mal, es posible que el sistema no arranque. Puede que sea aconsejable realizar una copia de seguridad antes de empezar siguiendo esta guía, <http://bit.ly/2gg5KGc>.

## Pasos comunes

Primero, instala un kernel estándar que soporte codificación MFC. Las instrucciones usadas aquí parten de @elatllat (<http://bit.ly/2gg82Fj>), pero @mdrjr también está trabajando en su propia versión (<http://bit.ly/2g2pVVc>) y yo también (<http://bit.ly/2gf0dfh>). Puedes usar mi configuración de kernel, que intenta añadir todos los módulos de HardKernel, o puedes modificarla para ajustarla a tus necesidades (<http://bit.ly/2gAr75I>). Otra posibilidad es utilizar

cualquier configuración por defecto que venga con el kernel que elijas.

El kernel no admite HMP (extensiones big.LITTLE) y trata a todos los núcleos como iguales. Esto se debe a que los parches HMP son muy inestables y pueden bloquear el sistema. En consecuencia, el rendimiento general del sistema será algo menor que cuando se utiliza el Kernel 3.10.

He añadido el procedimiento de compilación del kernel por comodidad, aunque deberías echar un vistazo al hilo oficial sobre compilación del kernel en <http://bit.ly/2fo18cv>, o revisar la guía de <http://bit.ly/1NVRprY>.

Primero, haz una copia de tu kernel, initrd, dtb y boot.ini:

```
$ cd /media/boot
$ sudo -i
# cp zImage zImage-3
# cp uInitrd uInitrd-3
# cp exynos5422-odroidxu3.dtb
exynos5422-odroidxu3-3.dtb
# cp boot.ini boot3.ini
```

Edita boot3.ini para apuntar a los archivos recién copiados añadiendo -3 al nombre de zImage, uInitrd y dtb. Al reemplazar el archivo boot.ini por este boot.ini modificado, podrás arrancar tu anterior kernel en caso de problemas.

Después, descarga el nuevo kernel:

```
$ sudo apt-get -y install bc curl \
\
gcc git libncurses5-dev lzop \
```

```
make u-boot-tools dos2unix
$ git clone --depth 1 \
-b odroidxu4-mihailescu2m-4.8 \
https://github.com/Dmole/linux.
git linux
$ cd linux
$ make odroidxu4_defconfig
```

Opcionalmente, puede conseguir mi config que soporta la mayoría de los periféricos USB (sintonizadores de TV, tarjetas de sonido, tarjetas wifi) y una gran cantidad de módulos de red (soporte LXC, VLANs, iptables):

```
$ wget http://pastebin.com/
raw/7YnakKmP -O .config
$ dos2unix .config
```

A continuación, compila el kernel:

```
$ make menuconfig
$ make -j 8 zImage dtbs modules
$ kver='`make kernelrelease`'
$ sudo cp arch/arm/boot/zImage
arch/arm/boot/dts/exynos5422-
odroidxu[34].dtb /media/boot
$ sudo cp .config /media/boot/
config
$ sudo make INSTALL_MOD_STRIP=1
modules_install
$ sudo make firmware_install
$ sudo cp .config /boot/config-
${kver}
$ cd /boot
$ sudo update-initramfs -c -k
${kver}
$ sudo mkimage -A arm -O linux -T
```

```
ramdisk -a 0x0 -e 0x0 -n initrd.
img-${kver} -d initrd.img-${kver}
uInitrd-${kver}
$ sudo cp uInitrd-${kver} /media/
boot/uInitrd
```

Modifica /media/boot/boot.ini y carga exynos5422-odroidxu4.dtb en lugar de exynos5422-odroidxu3.dtb y guarda boot.ini. Cierra el sistema y desconecta los cables HDMI y de alimentación. Si no haces esto, cuando arranques con el nuevo kernel, no tendrás conexión USB3 ni red. Sólo es necesario solo cuando se cambia entre los Kernel 3.x y 4.x por lo que he visto. Asegúrate de volver a conectar los cables antes de encenderlo.

## Pasos cuando utilizamos FFmpeg

Ahora que el kernel está listo, compila/installa un ffmpeg personalizado que soporte codificación por hardware. Compila tu propio ffmpeg usando los siguientes comandos:

```
# debian build tools
$ sudo apt-get install build-essential fakeroot devscripts \
libchromaprint-dev librubberband-dev libjs-bootstrap
# get the patched ffmpeg version
$ git clone -b v4l2_m2m-3.0.2 \
--depth=1 \
https://github.com/mihaieluscu2m/
FFmpeg.git
# install ffmpeg's build dependencies (~190 packages)
$ sudo apt-get build-dep ffmpeg
# build ffmpeg as deb packages with no checks (some checks fail)
$ cd FFmpeg
$ DEB_BUILD_OPTIONS="nocheck" debuild -b -uc -us
```

Cuando la compilación haya finalizado, deberías tener 23 paquetes deb en el mismo directorio que FFmpeg. Puede descargar estos paquetes pre-compilados desde aquí: <http://bit.ly/2g2m2iZ>. Luego, instala los paquetes necesarios:

```
$ cd ../
$ sudo dpkg -i *.deb
```

Esto debería reemplazar cualquier sistema ffmpeg instalado que soporte codificación por hardware. Utilizamos ahora el codificador de hardware para la transcodificación. Dependiendo de lo que desees transformar con ffmpeg, podrás obtener mejor o peor tasa de frames. Por ejemplo, cambiar el fps de salida de un video acorta aproximadamente unos 20-30 fps el tiempo de codificación. Los siguientes ejemplos no llegan a cambiar los fps:

```
# encode video only, about 50fps
(max)

$ ffmpeg -i big_buck_bunny_720p_
h264.mov -acodec aac -vcodec h264
-b:v 2M -pix_fmt nv21 bbb.mp4
```

```
# encode video only, about 110fps
(max)

$ ffmpeg -i big_buck_bunny_720p_
h264.mov -codec:v copy -codec:a
none -bsf:v h264_mp4toannexb
-f rawvideo - | ffmpeg -r 24
-i - -an -vcodec h264 -b:v 2M
-profile:v 10 -pix_fmt nv21 bbb.
mp4
```

```
#encode video and audio, about
75fps (max)

$ ffmpeg -i big_buck_bunny_720p_
h264.mov -codec:v copy -codec:a
none -bsf:v h264_mp4toannexb -f
rawvideo - | ffmpeg -r 24 -i -
-i big_buck_bunny_720p_h264.mov
-map 0:v:0 -vcodec h264 -b:v 2M
-profile:v 10 -pix_fmt nv21 -map
1:a:0 -c:a:1 aac bbb.mp4
```

El codificador selecciona automáticamente h264\_v4l2m2m que hace la codificación por hardware:

```
Stream #0:0 -> #0:0 (h264 (native) -> h264 (h264_v4l2m2m))
Stream #0:1 -> #0:1 (ac3 (native) -> aac (native))
```

Puedes ver los siguientes errores:

```
[h264_v4l2m2m @ 0xf3fa0] H264
codec detected, init annexb converter
[h264_v4l2m2m @ 0xf3fa0] Device
path not set, probing /dev/video*
[h264_v4l2m2m @ 0xf3fa0] exynos-
gsc.1:m2m is not the one we want
[h264_v4l2m2m @ 0xf3fa0] exynos-
gsc.0:m2m is not the one we want
[h264_v4l2m2m @ 0xf3fa0] s5p-mfc-
dec is not the one we want
[h264_v4l2m2m @ 0xf3fa0] Could
not find a valid device
```

Si es así, asegúrate de que el usuario con el que estás ejecutando ffmpeg, forma parte del grupo de video:

```
odroid@odroid:~$ id
uid=1000(odroid) gid=1000(odroid)
groups=1000(odroid),4(adm),20(dialout),24(cdrom),27(sudo),30(dip),
44(video),46(plugdev),115(lpadmin),
116(lightdm)
```

El resultado de muestra en el siguiente:

```
adrianp@odroid:~> ffmpeg -i
Sintel.2010.720p.mkv -acodec aac
-vcodec h264 -b:v 2M -pix_fmt
nv21 sintel-encoded.mp4
ffmpeg version 3.0.2-1ubuntu4
Copyright (c) 2000-2016 the FFmpeg
developers
built with gcc 5.4.0 (Ubuntu/
Linaro 5.4.0-6ubuntu1~16.04.2)
20160609
configuration: --prefix=/
usr --extra-version=1ubuntu4
--toolchain=hardened --libdir=/
usr/lib/arm-linux-gnueabihf
--incdir=/usr/include/arm-linux-
gnueabihf --cc=cc --cxx=g++
```

```
--enable-gpl --enable-shared
--disable-stripping --disable-
decoder=libopenjpeg --disable-
decoder=libschroedinger --enable-
avresample --enable-avisynth
--enable-gnutls --enable-ladspa
--enable-libass --enable-lib-
bluray --enable-libbs2b --en-
able-libcaca --enable-libcdio
--enable-libflite --enable-lib-
fontconfig --enable-libfreetype
--enable-libfribidi --enable-
libgme --enable-libgsm --enable-
libmodplug --enable-libmp3lame
--enable-libopenjpeg --enable-
libopus --enable-libpulse --en-
able-librubberband --enable-
librtmp --enable-libschroedinger
--enable-libshine --enable-lib-
snappy --enable-libsoxr --en-
able-libspeex --enable-libssh
--enable-libtheora --enable-
libtwolame --enable-libvorbis
--enable-libvpx --enable-lib-
wavpack --enable-libwebp --en-
able-libx265 --enable-libxvid
--enable-libzvbi --enable-openal
--enable-opengl --enable-x11grab
--enable-libdc1394 --enable-li-
biec61883 --enable-libzmq --en-
able-frei0r --enable-chromaprint
--enable-libx264 --enable-ver-
sion3 --disable-doc --disables-
programs --disable-avdevice
--disable-avfilter --disable-
avformat --disable-avresample
--disable-postproc --disables-
swscale --enable-libopencore_am-
rnb --enable-libopencore_amrwb
--enable-libvo_amrwbenc
    libavutil      55. 17.103 / 55.
17.103
    libavcodec     57. 24.102 / 57.
24.102
    libavformat    57. 25.100 / 57.
25.100
    libavdevice     57.  0.101 / 57.
0.101
    libavfilter     6. 31.100 /  6.
31.100
    libavresample   3.  0.  0 /  3.
0.  0
    libswscale      4.  0.100 /  4.
0.100
    libswresample   2.  0.101 /  2.
0.101
    libpostproc    54.  0.100 / 54.
0.100
Input #0, matroska,webm, from
'Sintel.2010.720p.mkv':
```

```
Metadata:
```

```
fontconfig --enable-libfreetype
--enable-libfribidi --enable-
libgme --enable-libgsm --enable-
libmodplug --enable-libmp3lame
--enable-libopenjpeg --enable-
libopus --enable-libpulse --en-
able-librubberband --enable-
librtmp --enable-libschroedinger
--enable-libshine --enable-lib-
snappy --enable-libsoxr --en-
able-libspeex --enable-libssh
--enable-libtheora --enable-
libtwolame --enable-libvorbis
--enable-libvpx --enable-lib-
wavpack --enable-libwebp --en-
able-libx265 --enable-libxvid
--enable-libzvbi --enable-openal
--enable-opengl --enable-x11grab
--enable-libdc1394 --enable-li-
biec61883 --enable-libzmq --en-
able-frei0r --enable-chromaprint
--enable-libx264 --enable-ver-
sion3 --disable-doc --disables-
programs --disable-avdevice
--disable-avfilter --disable-
avformat --disable-avresample
--disable-postproc --disables-
swscale --enable-libopencore_am-
rnb --enable-libopencore_amrwb
--enable-libvo_amrwbenc
    libavutil      55. 17.103 / 55.
17.103
    libavcodec     57. 24.102 / 57.
24.102
    libavformat    57. 25.100 / 57.
25.100
    libavdevice     57.  0.101 / 57.
0.101
    libavfilter     6. 31.100 /  6.
31.100
    libavresample   3.  0.  0 /  3.
0.  0
    libswscale      4.  0.100 /  4.
0.100
    libswresample   2.  0.101 /  2.
0.101
    libpostproc    54.  0.100 / 54.
0.100
Input #0, matroska,webm, from
'Sintel.2010.720p.mkv':
```

Metadata:

```
encoder       : libebml
v1.0.0 + libmatroska v1.0.0
creation_time : 2011-04-24
17:20:33
Duration: 00:14:48.03, start:
0.000000, bitrate: 6071 kb/s
Chapter #0:0: start 0.000000,
end 103.125000
Metadata:
    title       : Chapter
01
    Chapter #0:1: start
103.125000, end 148.667000
Metadata:
    title       : Chapter
02
    Chapter #0:2: start
148.667000, end 349.792000
Metadata:
    title       : Chapter
03
    Chapter #0:3: start
349.792000, end 437.208000
Metadata:
    title       : Chapter
04
    Chapter #0:4: start
437.208000, end 472.075000
Metadata:
    title       : Chapter
05
    Chapter #0:5: start
472.075000, end 678.833000
Metadata:
    title       : Chapter
06
    Chapter #0:6: start
678.833000, end 744.083000
Metadata:
    title       : Chapter
07
    Chapter #0:7: start
744.083000, end 888.032000
Metadata:
    title       : Chapter
08
    Stream #0:0(eng): Video:
h264 (High), yuv420p(tv, bt709/
unknown/unknown), 1280x544, SAR
1:1 DAR 40:17, 24 fps, 24 tbr, 1k
tbn, 48 tbc
```

```

Stream #0:1(eng): Audio: ac3,
48000 Hz, 5.1(side), fltp, 640
kb/s

Metadata:
    title      : AC3 5.1 @
640 Kbps

Stream #0:2(ger): Subtitle:
subrip

Stream #0:3(eng): Subtitle:
subrip

Stream #0:4(spa): Subtitle:
subrip

Stream #0:5(fre): Subtitle:
subrip

Stream #0:6(ita): Subtitle:
subrip

Stream #0:7(dut): Subtitle:
subrip

Stream #0:8(pol): Subtitle:
subrip

Stream #0:9(por): Subtitle:
subrip

Stream #0:10(rus): Subtitle:
subrip

Stream #0:11(vie): Subtitle:
subrip

Codec AVOption preset (Configuration preset) specified for output
file #0 (sintel-encoded.mp4) has
not been used for any stream.
The most likely reason is either
wrong type (e.g. a video option
with no video
streams) or that it is a private
option of some encoder which was
not actually used for any stream.
File 'sintel-encoded.mp4' already
exists. Overwrite ? [y/N] y
[h264_v4l2m2m @ 0xf3fe0] H264
codec detected, init annexb con-
verter
[h264_v4l2m2m @ 0xf3fe0] Device
path not set, probing /dev/video*
[h264_v4l2m2m @ 0xf3fe0] exynos-
gsc.1:m2m is not the one we want
[h264_v4l2m2m @ 0xf3fe0] exynos-
gsc.0:m2m is not the one we want
[h264_v4l2m2m @ 0xf3fe0] Using
device /dev/video1
Output #0, mp4, to 'sintel-encod-
ed.mp4':

```

```

Metadata:
    encoder      :
Lavf57.25.100

Chapter #0:0: start 0.000000,
end 103.125000

Metadata:
    title      : Chapter
01

Chapter #0:1: start
103.125000, end 148.667000

Metadata:
    title      : Chapter
02

Chapter #0:2: start
148.667000, end 349.792000

Metadata:
    title      : Chapter
03

Chapter #0:3: start
349.792000, end 437.208000

Metadata:
    title      : Chapter
04

Chapter #0:4: start
437.208000, end 472.075000

Metadata:
    title      : Chapter
05

Chapter #0:5: start
472.075000, end 678.833000

Metadata:
    title      : Chapter
06

```

```

Chapter #0:6: start
678.833000, end 744.083000

Metadata:
    title      : Chapter
07

Chapter #0:7: start
744.083000, end 888.032000

Metadata:
    title      : Chapter
08

Stream #0:0(eng): Video:
h264 (h264_v4l2m2m) ([33][0][0]
[0] / 0x0021), yuv420p, 1280x544
[SAR 1:1 DAR 40:17], q=2-31, 2000
kb/s, 24 fps, 12288 tbn, 24 tbc

Metadata:
    encoder      :
Lavc57.24.102 h264_v4l2m2m

Stream #0:1(eng): Audio: aac
(LC) ([64][0][0][0] / 0x0040),
48000 Hz, 5.1(side), fltp, 341
kb/s

Metadata:
    title      : AC3 5.1 @
640 Kbps

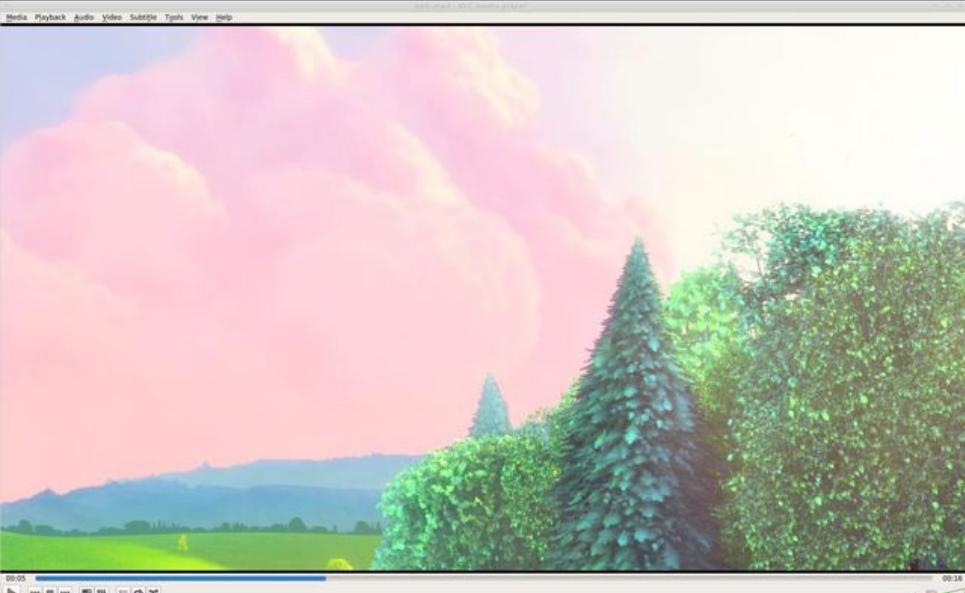
Metadata:
    encoder      :
Lavc57.24.102 aac

Stream mapping:
    Stream #0:0 -> #0:0 (h264 (na-
tive) -> h264 (h264_v4l2m2m))
    Stream #0:1 -> #0:1 (ac3 (na-
tive) -> aac (native))

Press [q] to stop, [?] for help

```

**Figura 1 - Demo de Big Buck Bunny**





# ODROID Magazine esta en Reddit!



## ODROID Talk Subreddit

<http://www.reddit.com/r/odroid>



```
[h264_v4l2m2m @ 0xf3fe0] Performing useless memcpy() on output pool because buffers do not match
[h264_v4l2m2m @ 0xf3fe0] This could be avoided by using av_v4l_buffer_pool_get_buffer() or av_v4l_buffer_pool_make_pipe()
[mp4 @ 0xb9c70] Timestamps are unset in a packet for stream 0. This is deprecated and will stop working in the future. Fix your code to set the timestamps properly
[mp4 @ 0xb9c70] Encoder did not produce proper pts, making some up.
[h264_v4l2m2m @ 0xf3fe0] No event occurred while waiting.
  0.1 bitrate=36881.3kbits/s
  speed=1.84x
  frame= 2027 fps= 44 q=-0.0
  Lsize= 383559kB time=00:01:25.20
  bitrate=36877.0kbits/s
  speed=1.84x
  video:379964kB audio:3551kB
  subtitle:0kB other streams:0kB
  global headers:0kB muxing overhead: 0.011593%
```

## Pasos cuando usamos gstreamer

Compila gstreamer desde la rama de memeka usando los comandos:

```
$ apt-get install gstreamer1.0-plugins-bad
$ git clone https://github.com/\mihaiulescu2m/gst-plugins-good
$ cd gst-plugins-good/
$ sudo apt-get install libgstreamer1.0-dev libgudev-1.0-dev \
libgstreamer-plugins-base1.0-dev \
dh-autoreconf automake autoconf \
libtool autopoint cdbs gtk-doc-tools libshout3-dev libaal-dev \
libflac-dev libdv4-dev libdv-dev \
libgtk-3-dev libtag1-dev \
libsoup2.4-dev gstreamer1.0-doc \
gstreamer1.0-plugins-base-doc
$ dpkg-buildpackage -us -uc -b
```

```
-j4
$ cd ../
$ sudo dpkg -i gstreamer*.deb
```

Como alternativa, puedes conseguir los paquetes pre-compilados desde <http://bit.ly/2gj7Iqm>. Para transcodificar algo, primero debe identificar la codificación y las interfaces de decodificación, puesto que cambian en cada inicio:

```
$ decoder=`gst-inspect-1.0 | grep v4l2 | grep videodec | cut -d ":" -f 2`
$ encoder=`gst-inspect-1.0 | grep v4l2 | grep h264enc | cut -d ":" -f 2`
$ gst-launch-1.0 filesrc location=big_buck_bunny_720p_h264.mov ! qtdemux ! h264parse ! $decoder ! $encoder extra-controls="encode,h264_level=10,h264_profile=4,frame_level_rate_control_enable=1,video_bitrate=2097152" ! h264parse ! matroskamux ! filesink location=bbb.mkv
```

Para comentarios, preguntas y sugerencias, visita el post original en <http://bit.ly/2g0vnsn>.

**¡Tu perro estará satisfecho con la codificación por hardware de tu ODROID-U4!**



# COMO UTILIZAR UN ODROID COMO ALTAVOZ BLUETOOTH

## MUSICA PARA LAS MULTITUDES

por @codewalker

**S**i tienes un ODROID instalado y configurado como un PC para el coche, puede que quieras guardar tu colección de música en un dispositivo portátil, como un teléfono o una tablet, y que éste se conecte automáticamente a los altavoces del vehículo cuando conduces. O, puede que tengas un Shield HiFi conectado a tu ODROID-C2 junto con un sistema de sonido estéreo de alta gama, y quieras reproducir música desde el teléfono inteligente de un amigo a través de éste. Siguiendo los pasos de este artículo, puedes utilizar tu ODROID como altavoz bluetooth.

Para empezar, asegúrate de que tiene un dongle bluetooth conectado a tu ODROID, como se muestra a continuación.



ODROID con dongle Bluetooth en el puerto USB

A continuación, abre la aplicación Terminal Emulator en tu ODROID, como aparece a continuación:



Iniciando Terminal emulator

Después, cambie al usuario raíz y confirma el cuadro de diálogo que aparece:

```
$ su
```

Vuelve a montar el sistema de archivos raíz con privilegios de lectura/escritura para que puedas realizar cambios:

```
# mount -o rw,remount /
```

Ejecutando el comando mount

Utiliza un editor de texto como vi para editar el archivo "build.prop":

```
# vi /build.prop
```

Editando el archivo build.prop utilizando vi

Tras cargar el archivo, agrega las dos líneas siguientes al final, guarda el archivo y reinicia el sistema:

```
#Bluetooth Audio A2dp Sink
persist.service.bt.a2dp.sink=true
```

Tras reiniciar el ODROID, activa la opción " Media Audio ", como se muestra a continuación:



Activando la opción Media Audio

Por último, conecta el smartphone o la tablet al ODROID asociándolo a través de bluetooth y reproduzca tu música. Para comentarios, preguntas y sugerencias, visita la post original en <http://bit.ly/2f2d08H>.

# KERNEL EN TIEMPO REAL DE ODROID-C2

## EMPECEMOS CON RTOS

por Anand Moon (@moon.linux)

**U**n sistema operativo en tiempo real (RTOS) es un sistema operativo destinado a servir en tiempo real los datos resultantes del procesamiento de las aplicaciones a medida que se van generando, normalmente sin retrasos de almacenamiento en búfer. El kernel estándar de Linux sólo cubre algunos requisitos en tiempo real al proporcionar operaciones POSIX básicas para la gestión del tiempo del espacio de usuario, pero no garantiza plazos de tiempo difíciles. Si aplicamos el parche Preemption en tiempo real (RTPreempt) de Ingo Molnar y la capa genérica de eventos de reloj de Thomas Gleixner con soporte de alta resolución, el kernel consigue funciones completas de tiempo real.

El parche RT-Preempt ha despertado mucho interés en toda la industria. Su diseño limpio y consecuente objetivo hacia una integración estándar lo convierte en una opción muy interesante para complejas y sólidas aplicaciones en tiempo real. No es ninguna sorpresa ver aplicaciones que van desde el audio profesional al control industrial usando RT Linux.

### Casos de uso

- Hacer que las primitivas (elementos básicos de los programas) bloqueadas en el núcleo (usando spinlocks) sean pre-emplazables aunque se reimplementen con rtmutexes.
- Las secciones críticas protegidas por spinlock\_t y rwlock\_t son ahora pre-emplazables.

- Implementación de legado de prioridad para rw\_semaphores, spinlocks y mutex dentro del kernel.
- Conversión de manipuladores de interrupción en subprocesos del kernel.
- Conversión de la antigua API reloj Linux en infraestructuras separadas para temporizadores de kernel de alta resolución más uno para tiempos de espera.
- Mejoras en el reloj que da ligar a relojes POSIX del espacio de usuario con alta resolución.

### Desarrollando el Kernel

Para desarrollar un kernel Linux en tiempo real (RT), necesitamos actualizar el kernel con parches de kernel en tiempo real. El archivo de parches RT lo puedes encontrar en <http://bit.ly/2g3MiJ2>, y los últimos parches para el kernel 3.14.x los puedes localizar en <http://bit.ly/2goVUSQ>. Puede descargar el árbol de fuentes del kernel de Linux para el ODROID-C2 desde <http://bit.ly/2fNFOi4>. Los parches RT deben coincidir con la versión del kernel de Linux, de modo que debes elegir la serie de parches correcta.

```
$ git clone --depth 1 \
-b odroidc2-3.14.y \
https://github.com/hardkernel/
linux.git \
odroidc2-3.14.y-rt
$ cd odroidc2-3.14.y-rt
$ wget \
```

```
https://www.kernel.org/pub/
linux/kernel/\ 
projects/rt/3.14/older/\ 
patch-3.14.65-rt68.patch.gz
$ gunzip patch-3.14.65-rt68.
patch.gz
$ patch -p1 < \
patch-3.14.65-rt68.patch
```

El kernel RT ARM64 no tiene algunos parches, así que tenemos que coger el kernel 3.18.y y aplicar esos parches también. Una vez que se apliquen los parches adicionales, podemos desarrollar el kernel. Los parches necesarios se pueden encontrar en la serie de parches 3.18.y-rt:

arm64: Mark PMU interrupt IRQF\_NO\_THREAD.patch

arm64: Allow forced irq threading.patch

arch/arm64: Add lazy preempt support.patch

arm64: replace read\_lock to rcu lock in call\_step\_hook

Algunos archivos tienen conflictos, de modo que hay que actualizarlos manualmente antes de crear el kernel. Puede encontrar los enlaces a todos los parches actualizados en el siguiente repositorio en <http://bit.ly/2g6R3Di>.

### Compilando el kernel

La compilación se hace con make. La argumento -j del comando make acelerará la compilación:

```
$ make -j4 Image dtbs modules
```

Una vez finalizado el proceso, tendrás un kernel Linux compilado (imagen), el archivo árbol de dispositivos (.dtb) y los módulos del kernel (.ko). En los siguientes pasos suponemos que tu lector de tarjetas de memoria USB está asignado a /dev/sdc. ¡Tenlo presente y comprueba cómo está montada tu tarjeta!

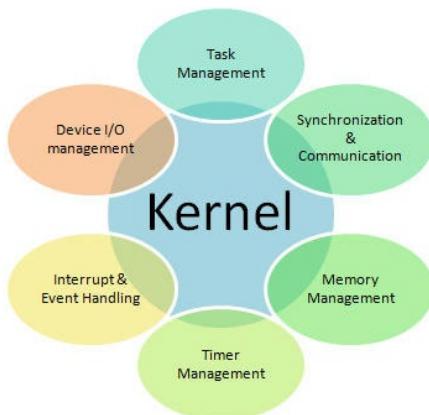
En primer lugar, inserta el soporte de arranque, ya sea el módulo eMMC o la tarjeta SD en el lector de tarjetas USB y conecta el lector a tu PC Linux. A continuación, copie la imagen y DT (meson64\_odroidc2.dtb) en la primera partición (FAT) del soporte de arranque:

```
$ mkdir -p mount
$ sudo mount /dev/sdc1 ./mount
$ sudo cp arch/arm64/boot/Image\ arch/arm64/boot/dts/meson64_
odroidc2.dtb\.
./mount && sync && sudo umount
./mount
```

Finalmente, copia los módulos driver a la 2<sup>a</sup> partición (EXT4) del soporte de arranque:

```
$ sudo mount /dev/sdc2 ./mount
$ sudo make modules_install \
ARCH=arm64 \
INSTALL_MOD_PATH=./mount && sync \
&
& sudo umount ./mount
$ rm -rf mount
```

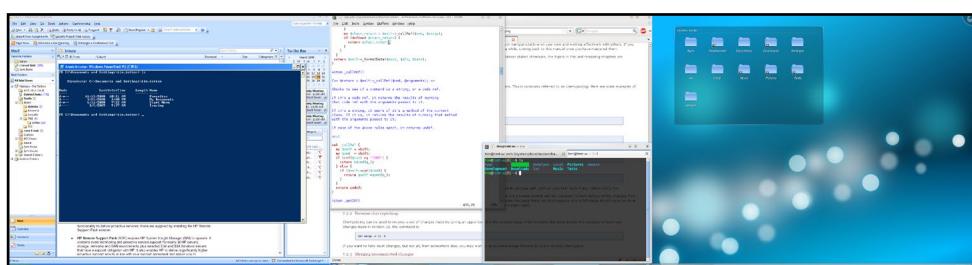
Para comentarios, preguntas o sugerencias, visita el post original en <http://bit.ly/2f2b0s8>.



# DOMINANDO EL MUNDO CON SYNERGY

## CRONICAS DE UN CIENTIFICO LOCO

por Bo Lechnowsky (@respectech)



Ejecutar incluso entornos heterogéneos X86 Windows x ARM Linux

**S**i tuviera un subordinado, no de tipo animado sino del tipo “Pull the Switch”, lo miraría con asombro mientras que lanzaría ventanas en múltiples monitores ejecutadas por múltiples ordenadores, copiando y pegando datos como un loco. Sabes que como científico con planes para dominar el mundo, no puedes aflojar la marcha por la latencia al lanzar aplicaciones y compartir datos entre ellas. Es por ello que instalaste Synergy en primer lugar, tal y como se describe en la edición de junio de 2016 de ODROID Magazine (<http://bit.ly/1xxSbRw>).

Recientemente, has decidido desempolvar un venerable ODROID-U2 para añadirle a Synergy. Sin embargo, no querías volver a compilar Synergy desde cero. Actualizaste el sistema con los siguientes comandos:

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

A continuación, descargaste la versión compilada para el ODROID-XU4 de <http://bit.ly/22lNiL>, pero tras descomprimir y mover los archivos a la carpeta /usr/bin, apareció el siguiente mensaje:

```
synergyc: /usr/lib/arm-linux-gnueabihf/libstdc++.so.6: version `GLIB-
CXX_3.4.20' not found (required by synergyc)
```

Tras una búsqueda por Internet, encuentras una fácil solución a este problema:

```
$ sudo add-apt-repository ppa:ubuntu-toolchain-r/test
$ sudo apt update
$ sudo apt install g++-4.9
```

Una vez configurado el U2 usando el botón “Configure Server...” en Synergy Server que se ejecuta en tu máquina principal y ejecutando el comando synergyc en el U2, todo lo que necesitabas era hacer “Parar” e “Iniciar” Synergy Server y tu U2 entraría en funcionamiento. ¡La vida es buena, y la dominación del mundo está un paso más cerca!

# ¿POR QUE AÑADIR REFRIGERACION LIQUIDA A UN ODROID-XU3 O XU4?

por Michael Lee Wood (@mlwood37)



**E**s muy conocido por toda la comunidad ODROID que la temperatura del XU4 y XU3 empieza a dispararse con cargas de trabajo muy pesadas. Esto da origen rápidamente al llamado estrangulamiento térmico, donde la placa se ralentiza automáticamente tras alcanzar una cierta temperatura para prevenir posibles daños a la misma. El XU3 y XU4 tiende a bajar la velocidad del reloj desde los 2 GHz a 900 MHz con el objeto de enfriarse durante un rato. Sin embargo, si mejoramos la capacidad que tiene el ODROID para refrigerarse, éste puede mantener su máximo rendimiento durante más tiempo. Hay muchas formas de mejorar la refrigeración, incluyendo un dissipador de calor más grande con un ventilador de 40 mm, no obstante la refrigeración líquida tiene dos ventajas clave: tiene una capacidad térmica mucho mayor y es más silenciosa, ya que la mayoría de los kits de refrigeración líquida del mercado utilizan ventiladores más grandes que son menos ruidosos que el típico ventilador que viene de fábrica con el XU3 o XU4.

Si estás considerando la posibilidad de utilizar refrigeración líquida en el ODROID, necesitas saber cómo lo vas a refrigerar y qué hardware vas a utilizar. Existen muchos y muy diferentes tipos de componentes de refrigeración personalizados en el mercado y si no tienes cuidado, tu configuración de refrigeración puede costarte incluso más que la pro-

pia placa. Cualquier sistema de refrigeración por agua necesita tres componentes principales: un bloque de agua que se acopla al chip de tu ODROID para transferir el calor, un radiador y un ventilador para enfriar el agua, y una bomba que permite que el agua circule a través del sistema. En mi particular desarrollo, he optado por una Corsair Hydro Series H45 por muchas razones, una de ellas es que la bomba está integrada en el propio radiador. Muchos sistemas de refrigeración líquida AIO (Todo en Uno) tienen la bomba encima del sistema de refrigeración de la CPU. Esto no nos sirve a nosotros, puesto que no estamos instalando el sistema en un PC estándar. Tener la bomba en el radiador nos permite mantener el sistema lo más compacto posible, al mismo tiempo que nos resulta más fácil cortar el bloque de la CPU y conectar nuestro propio bloque de refrigeración líquida adaptado a nuestro ODROID.

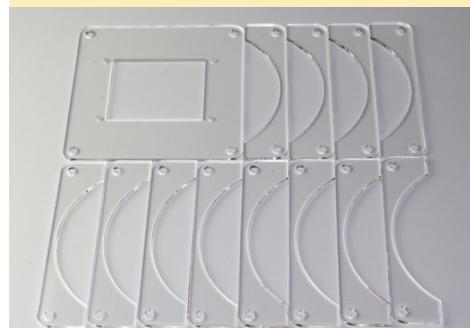
**Figura 1 - Resultado final del proyecto de refrigeración líquida**



La mayoría de los sistemas de refrigeración por agua para CPU no tienen un bloque de agua compatible con tu ODROID dado su tamaño, así que necesitarás encontrar uno que encaje en un chip North Bridge, que es similar en tamaño al SoC de nuestro ODROID. En esta cuestión no quería gastar dinero, ya que tenía un viejo bloque de agua EKWB por ahí sin hacer nada. No era la solución perfecta, ya que necesitaba recortarlo para que encajara en el chip del XU4. Chaflané el cobre de modo que, una vez acoplado, no tocaba los puertos HDMI o de sonido. ¡Lo último que queremos en nuestro XU4 es un cortocircuito! También tuve que cortar una sección del bloque para evitar el puerto de salida de sonido. Hay algunos bloques en el mercado que encajan perfectamente, sin embargo, pueden llegar a costar alrededor de 35.00£ (43\$) más gastos de envío y embalaje, y que probablemente no incluirá el resto del sistema de refrigeración.

Para intentar mantener las cosas lo más compactas posibles, también acople

**Figura 2 - Las piezas de acrílico necesarias para montar tu ODROID-XU4**



mi ODROID al radiador usando algunos tornillos y piezas de plástico personalizadas cortadas con láser. Puedes conseguir estas piezas, como las que se muestra en la Figura 2, recurriendo a cualquier persona que disponga de un cortador láser decente y utilizando los planos que he puesto a disposición en <http://bit.ly/2fy3llS>. Mi desarrollo también incluye una adaptación con un taladro y un tornillo autorroscante para acceder al refrigerante y reducir el tamaño del tubo para ajustarlo a mi tamaño.

## Herramientas y piezas

Ahora, vamos a pasar a los pormenores para montar este sistema refrigerado por agua, incluyendo los pasos que llevaré a cabo para cortar el bloque de agua al tamaño deseado. En primer lugar, aquí tienes una lista de las herramientas que necesitarás:

- Destornilladores
- Herramienta rotativa Dremel con herramientas de corte de disco.
- Acceso a un cortador láser o una impresora 3D
- Segueta
- Papel de lija, una amoladora o cualquier cosa que pueda usar para moler el bloque de cobre.
- Taladro con una broca con el mismo tamaño que el tornillo autorroscante.
- Stanley knife or Tube cutters
- Jeringa para llenar el AIO con refrigerante nuevot

Estas son las piezas que utilizaremos:

- ODROID XU4 o XU3 de Hardkernel
- Corsair Hydro H45 (número de serie CW-9060028-WW)
- Bloque de refrigeración por agua
- Adaptador recto John Guest Rosca de 3/8" con conexión de tubo de 1/4" NPTF
- 0,25 metros de polietileno lineal de baja densidad de 10 mm

- Acrílico de 3mm claro o coloreado (para mecanizar tus piezas)
- Fuente de alimentación de 12v específicamente para la bomba y ventilador de 120mm.
- Tornillos/tuercas para todo.
- Pasta térmica (se recomienda MX4)
- Refrigerante Mayhems XT1 + agua DI
- Tornillo autorroscante y arandela de caucho

## Proceso de montaje

En primer lugar, tendrás que comprobar el acoplamiento del bloque de agua que has seleccionado para tu ODROID. Cuando realices las pruebas, asegúrate de que no esté cerca del puerto HDMI ni de los puertos de sonido. Puede que tengas que quitar la parte superior y cortar y achaflanar el bloque de cobre para que encaje perfectamente. Una vez hecho esto, pon el bloque de agua y tu ODROID a un lado.

A continuación, coge el Adaptador John Guest Straight 3/8" con tuvo de conexión 1/4" y corta los extremos con tu segueta para que enrosquen en tu bloque de agua. Si es necesario, utiliza cinta PTFE alrededor de las piezas para que queden bien sellados. Asegúrate de que las piezas estén bien apretados y que el sellado tenga solidez para evitar fugas.

Después, coge tu tubo de 10 mm de polietileno de baja densidad y cortarlo a medida, acóplalo el extremo del adaptador John Guest. Asegúrate de que está bien apretado y encajado. Aquí es donde tienes que empujar tu AIO en el bloque.

Tras completar este paso, estamos listos para conectar nuestro Corsair Hydro H45 al bloque de agua. Corta los tubos más cercanos al bloque de refrigeración de agua que viene conectado a tu sistema AIO. Deberías dejar un poco de tubo extra en caso de que necesites usarlo de nuevo en el futuro. Coge un recipiente y recoge el refrigerante que hay dentro de tu sistema refrigerador Corsair. Depende de ti el reutilizar este refrigerante, pero



**Figura 3 - El tornillo autorroscante y el orificio para llenar el refrigerante**

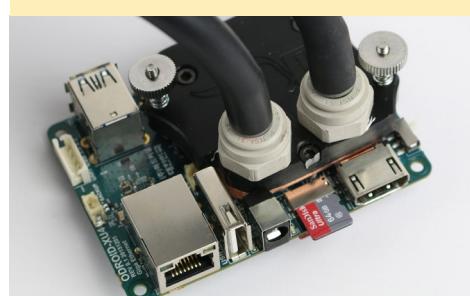
no es un refrigerante de muy alta calidad y probablemente esté contaminado con manchas de aluminio.

A continuación, prepararemos nuestro radiador para recibir el refrigerante, puesto que ha sido diseñado como un sistema de circuito cerrado. Coge tu taladro y poco a poco perfora la parte superior (en el lado superior derecho es el mejor sitio para hacerlo) del radiador Corsair Hydro H45 y si es posible, hazlo al revés para que no entre partículas de metal en el radiador. Una vez que haya hecho esto, coge un pequeño tornillo autorroscante y agrega una arandela de goma al mismo. Ahora atornilla tu tornillo autorroscante con cuidado, asegúrate de no apretarlo demasiado.

Utilizando los planos a los he hecho referencia anteriormente, corta las aletas de acrílico de 3 mm y la lámina de montaje para el XU4/XU3. Lo puedes hacer en el color que quieras. Quien lo desee puede hacer una impresión en 3D o hacerlos de madera. Utiliza este artículo como referencia para realizar las modificaciones que quieras.

A continuación, acoplaremos el bloque de agua al AIO. Analiza cómo encajará todo y corta los tubos AIO tanto

**Figura 4 - Un vistazo de cerca al montaje del bloque de agua**



# MINI METRO

## UN JUEGO PERFECTO PARA PREGUNTARSE POR LA ESPERA DE TU METRO

por Bruno Doiche

**T**omar el metro para ir a los lugares no es algo que suela mantener tu mente ocupada, pero hay un cierto sentido en esto cuando juegas a este juego mientras esperas a que llegue el metro para llevarte a casa al final de un día de duro trabajo. Si tienes buenos recuerdos de haber jugado Sim-City, ¡Este juego será tu próximo mejor despilfarro de tiempo al mejor estilo posible!



Para ser un juego que tenía todas las posibilidades de ser algo muy trivial, realmente te enganchará.



<https://play.google.com/store/apps/details?id=nz.co.codepoint.minimetro>

como te sea posible. Una vez hecho esto, retira el tornillo autorroscante y utiliza una jeringa para llenar el radiador con tu Mayhems XT1 (un refrigerante no tóxico que no corroa el cobre o aluminio). Tómate un tiempo para llenar bien el radiador, ya que necesitas sacar tanto aire como te sea posible del circuito. Inclina y mueve el AIO mientras lo llenas para ayudarte a eliminar el aire. Puedes encender la bomba después de cada llenado para extraer aire atrapado en la misma. Sin embargo, no enciendas la bomba si no tiene refrigerante, ya que la puedes dañar.

Una vez llenado el refrigerante, vuelve a poner el tornillo autorroscante con la arandela de goma. Prueba la bomba sin colocar nada para ver si existen fugas. Haz esto al menos 1 a 2 horas antes de continuar para asegurarte de que todo está correctamente instalado.

Ahora está listo para colocar las aletas de plástico y el bloque de agua en el frontal del AIO. Añade pasta térmica a tu SoC. Un pegote del tamaño de un guisante es más que suficiente. Tómate tu tiempo y recuerda que cuando ajustes el bloque de agua no aprietas demasiado los tornillos de refuerzo al XU4/XU3 ya que puedes doblar la PCB, lo cual puede echar a perder el ODROID. Luego, simplemente superpón las aletas en el Corsair AIO y atorníllalo todo. No olvide montar el XU4 en la lámina base más grande, puede que quieras utilizar las arandelas de elevación que he añadido a los planos (formato DXF).

Figura 5 - Otro ángulo del diseño final

El ODROID no tiene suficiente potencia para alimentar el sistema de refrigeración por agua, de modo que he tenido que utilizar una fuente de alimentación de 12 voltios y 5 amperios, que está conectado a un regulador de corriente/voltaje 5-30v 5A con 2 pantallas (<http://ebay.eu/2gfpuuZ>). Conectamos la bomba y el ventilador directamente a la fuente de alimentación de 12V, que coge 12 voltios y 0.45 amperios, y luego agregamos el regulador al circuito para bajar la tensión a 5 voltios y a 4 amperios, que podemos utilizar para alimentar el ODROID-XU4. Esto nos permite alimentar todo nuestro sistema desde una única toma de corriente.

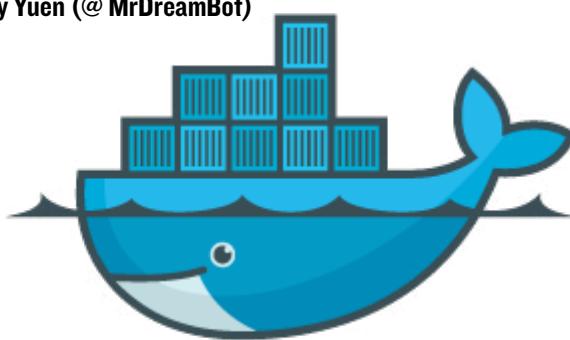
Por último, pero no menos importante: ¡Ponlo a prueba! Utiliza una herramienta de testeо para CPU para ver cómo van las cosas. Puede ajustar suavemente los tornillos del bloque de agua para conseguir una refrigeración óptima. No los apriete excesivamente ya que puedes doblar la PCB. Si quieres, también puedes añadir una abrazadera en la parte posterior de la placa para ayudarte a detener cualquier posible torcedura de la PCB. ¡Buena suerte y disfruta de tu ODROID con el máximo rendimiento!



# DOCKER 101

## PARTE 2 - MODO SWARM

por Andy Yuen (@MrDreamBot)



# docker

**E**n la Parte 1 de este tutorial, tratamos el clásico “`docker run`” y otros comandos. En la Parte 2, vamos a aprender los comandos del modo swarm que son nuevos en la versión 1.12.X de Docker. Entonces, ¿Qué es el modo Swarm y por qué lo necesitamos?

Todos los comandos de Docker que analizamos en la Parte 1 ejecutan los contenedores de Docker en una máquina local. Hay un límite en el número de contenedores que se pueden ejecutar en una única máquina debido a las limitaciones de la CPU y la memoria. Y cuando esa máquina falla, todas las aplicaciones que se estén ejecutando no estarán disponibles. Para proporcionar alta disponibilidad, escalabilidad, organización y gestión, se necesita un entorno de clúster. El modo Swarm es el entorno de clúster integrado (swarm) para motores Docker, aunque existen otros motores de organización para Docker (por ejemplo, Kubernetes).

bajo llamadas tareas a los nodos operarios y realiza planificaciones, funciones de administración y mantenimiento de determinados estados de un swarm. Puede haber varios nodos de gestión, pero siempre es un número impar de nodos de gestión debido al uso del protocolo Raft Consensus (<https://raft.github.io/>). Un nodo Gestor también puede ser un nodo Operativo al mismo tiempo. El almacén de estado guarda información sobre el estado del clúster y la configuración definida por el usuario. La información está organizada en objetos tales como clúster, nodos, servicios, tareas y redes.

Nodo Operativo, que ejecuta tareas recibidas de un nodo Gestor

La interacción del usuario con el swarm se hace a través de servicios. Un servicio es la definición de las tareas que se deben ejecutar en los nodos operativos. El escalado se logra utilizando un modelo de réplica de servicios en el que se ejecuta un número específico de tareas en los nodos operativos para satisfacer el estado deseado de réplicas objetivo que se van a ejecutar. El swarm se administra mediante una interfaz de línea de comandos a través del nodo gestor. En las siguientes secciones, trataremos cómo crear un swarm, los servicios y asignar el estado deseado. Si deseas llevar a cabo los ejercicios de este tutorial, tendrá que cumplir con los siguientes requisitos previos:

### Requisitos previos

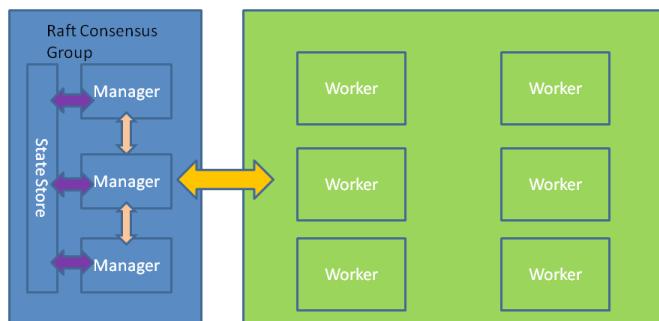
Para seguir este tutorial, debe tener lo siguiente:

Dos o más dispositivos ODROID-C2 conectados al mismo switch Ethernet al que está conectado tu PC. La siguiente Figura 2a muestra mi configuración, que consta de 5 ODROID-C2 y mi portátil, todos conectados en un único entorno de red. Mis nodos ODROID-C2 y sus roles se resumen en la Figura 2b.

Su PC debe ser capaz de conectarse vía SSH al nodo gestor c2-swarm-00, desde el cual se administra el swarm.

Tus nodos deben ejecutar la versión 1.12.X de Docker. Para aquellos que utilicen un sistema operativo cuyo repositorio de

### Docker Swarm Architecture

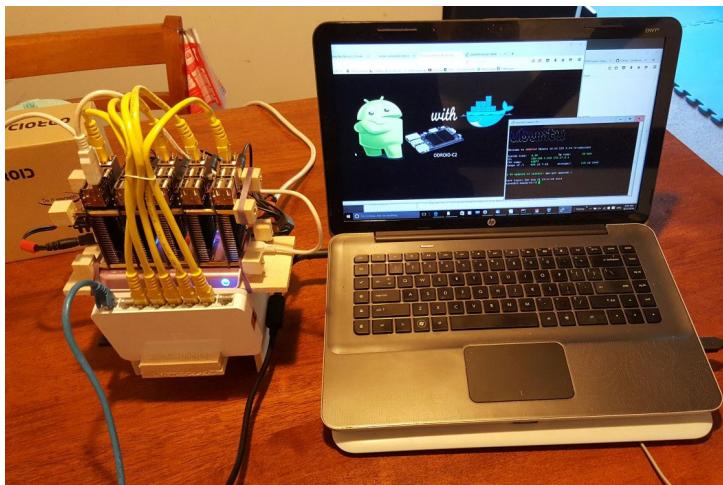


La arquitectura Swarm

El modo Swarm consiste en tener múltiples instancias de motores Docker, llamadas nodos. Hay dos tipos de nodos:

Nodo Gestor o Manager, que distribuyen unidades de tra-

# MODO SWARM DE DOCKER



```
ayuen@c2-swarm-00: ~
Network: null bridge host overlay
Swarm: active
NodeID: 8jw6y313hmt3vfafmeidinro
Is Manager: true
ClusterID: 5ioxna6w5y90qzd3x3db8anh6
Managers: 1
Nodes: 5
Orchestration:
  Task History Retention Limit: 5
Raft:
  Snapshot interval: 10000
  Heartbeat tick: 1
  Election tick: 3
Dispatcher:
  Heartbeat period: 5 seconds
CA configuration:
  Expiry duration: 3 months
Node Address: 192.168.1.100
Runtimes: runc
Default Runtime: runc
Security Options: apparmor seccomp
Kernel Version: 3.14.72-odroidc2
Operating System: Ubuntu 16.04 LTS
OSType: linux
```

## El modo Swarm de un vistazo

```
$ docker swarm join \
--token tokengoeshere \
192.168.1.100:2377
```

Tu único será devuelto desde el comando init del swarm de docker que ejecutamos anteriormente. Se trata de un mecanismo de seguridad que garantiza que sólo los nodos válidos pueden unirse al swarm. Ten en cuenta que sólo tiene que hacer esto una vez ya que la configuración se guarda en el almacenamiento de estado, que se mantiene aunque reiniciemos.

Solo necesitamos un nodo Gestor para hacer que el swarm funcione. En este tutorial, no vamos a agregar nodos gestores adicionales. Para ver el resultado, emite el siguiente comando desde el Gestor:

```
$ docker node ls
```

Luego echamos un vistazo a lo que este comando muestra en una típica lista.

Si ha perdido el token antes del proceso de init, no te preocupa, puede recuperarlo introduciendo el siguiente comando desde el nodo gestor. Para obtener el token de un gestor, reemplaza "worker" en el comando por "manager":

```
ayuen@c2-swarm-00: ~
Default Runtime: runc
Security Options: apparmor seccomp
Kernel Version: 3.14.72-odroidc2
Operating System: Ubuntu 16.04 LTS
OSType: linux
Architecture: aarch64
CPUUs: 4
Total Memory: 1.679 GiB
Name: c2-swarm-00
ID: 7DLX:G4LS:2CCZ:51DD:6UQH:D2M4:IRBQ:CTF7:S5WG:IDYK:X23Q:4YHN
Docker Root Dir: /media/usbdrive/docker
Debug Mode (client): false
Debug Mode (server): false
Registry: https://index.docker.io/v1/
Insecure Registries:
  127.0.0.0/8
ayuen@c2-swarm-00:~$ docker node ls
ID           HOSTNAME   STATUS  AVAILABILITY  MANAGER STATUS
17o9wmbv4obov9rjoy8x14rt  c2-swarm-02  Ready   Active
7obh8e9rjvo17rd68h73akfmt  c2-swarm-03  Ready   Active
8jw6y313hmt3vfafmeidinro * c2-swarm-00  Ready   Active      Leader
azihbc7npwizhes53uoxzomsx  c2-swarm-04  Ready   Active
ciugdv2xacb6bpff5gljqcqr5  c2-swarm-01  Ready   Active
ayuen@c2-swarm-00:~$
```

## Una lista de nodos docker disponibles

```
$ docker swarm join-token worker
```

Cuando quieras que un nodo se desvincule del swarm, introduce el siguiente comando desde ese nodo:

```
$ docker swarm leave
```

Por defecto, el nodo gestor también es un nodo operativo, si no quieres que el gestor ejecute ninguna tarea de servicio, puedes “vaciar” el gestor del siguiente modo:

```
$ docker node update --availability drain c2-swarm-00
```

De hecho, puedes utilizar el comando drain en cualquier nodo operativo. Si el nodo operativo ejecuta tareas de servicio cuando utilizas el comando drain, se cerrarán esas tareas y dejará que el swarm las inicie en otros nodos operativos para cumplir con el estado deseado. Cuando desees que el nodo ejecute tareas de servicio de nuevo, utiliza el siguiente comando en el nodo gestor:

```
$ docker node update --availability active c2-swarm-00
```

## Creando un servicio

Ahora que nuestro swarm está funcionando, vamos a crear nuestro primer servicio para hacer ping al nodo gestor. Podemos ver esto en la siguiente imagen. Escribe los siguientes comandos desde nodo gestor:

```
$ docker service create --replicas 1 --name ping-service mrdreambot/arm64-busybox-httpd /bin/ping 192.168.1.100
$ docker service ls
$ docker service inspect --pretty pingservice
```

Podemos ver que sólo hay 1 instancia de servicio funcionando. Escribe el siguiente comando:

```
$ docker service ps pingservice
```

Nos dice que el servicio se está ejecutando en c2-swarm-02. Inicia un terminal hacia c2-swarm-02 y escribe esto:

```
$ docker ps
$ docker logs pingservice.1.eic3ca0o4h0gxr675ncveptv
```

Como referencia, pingservice.1.eic3ca0o4h0gxr675ncveptv es el ID del contenedor docker identificado en el resultado del comando docker ps. Los registros log, como se muestra a continuación, muestran que pingservice está haciendo ping a

```

ayuen@c2-swarm-00: ~
Welcome to ARMbian Ubuntu 16.04 LTS 3.14.72-odroidc2
System load: 1.67 Up time: 16 sec
IP: 172.19.0.1 172.18.0.1
CPU temp: 43°C
Usage of /: 56% of 7.1G

[ 13 updates to install: apt-get upgrade ]

Last login: Sat Oct 29 05:39:13 2016 from 192.168.1.139
ayuen@c2-swarm-00:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS
53k68m2dyg76wtkmn8m9t4 c2-swarm-03 Ready Active
53l1upc0n482wrefye74pmot c2-swarm-02 Down Active
8dm1k30449c505e76057q21 c2-swarm-04 Down Active
9ipf4t7nfp5emifey9glidip * c2-swarm-00 Ready Active Leader
d3mzrnbab6swiwhoyryk c2-swarm-01 Ready Active
ayuen@c2-swarm-00:~$ docker service create --replicas 1 --name pingservice mrdreambot/arm64-busybox-httpd /bin/ping 192.168.1.100
amqrqd7w2d971sr8p6pkq0
ayuen@c2-swarm-00:~$ docker service ls
ID NAME REPLICAS IMAGE COMMAND
amqrqd7w2d971sr8p6pkq0 pingservice 1/1 mrdreambot/arm64-busybox-fish
ayuen@c2-swarm-00:~$ docker service inspect --pretty pingservice
ID: amqrqd7w2d971sr8p6pkq0
Name: pingservice
Mode: Replicated
Replicas: 1
Placement:
UpdateConfig:
Parallelism: 1
On failure: pause
ContainerSpec:
Image: mrdreambot/arm64-busybox-httpd
Args: /bin/ping 192.168.1.100
Resources:
ayuen@c2-swarm-00:~$
```

## Proceso de creación de servicios

mi gestor 192.168.1.100.

Para modificar la escala para ejecutar 5 instancias, ejecuta el comando desde el gestor swarm:

```
$ docker service scale pingservice=5
```

```

ayuen@c2-swarm-02: ~
Using username "ayuen".
Authenticating with public key "imported-openssh-key"
ayuen@c2-swarm-02: ~
Welcome to ARMbian Ubuntu 16.04 LTS 3.14.72-odroidc2
System load: 0.22 Up time: 8 min
IP: 172.17.0.1 172.18.0.1
CPU temp: 43°C
Usage of /: 71% of 7.1G

[ 76 updates to install: apt-get upgrade ]

Last login: Sat Oct 29 21:26:22 2016 from 192.168.1.139
ayuen@c2-swarm-02:~$ docker ps
CONTAINER ID IMAGE PORTS NAMES
6ccac35475bb mrdreambot/arm64-busybox-httpd:latest "/bin/busybox /bin/ping" 5 minutes ago Up 5 minutes
pingservice.1.eic3ca0o4h0gxr675ncveptv

ayuen@c2-swarm-02:~$ docker service ps pingservice.1.eic3ca0o4h0gxr675ncveptv | less
PING 192.168.1.100 (192.168.1.100) 56 bytes
64 bytes from 192.168.1.100: seq=0 ttl=63 time=1.222 ms
64 bytes from 192.168.1.100: seq=1 ttl=63 time=1.087 ms
64 bytes from 192.168.1.100: seq=2 ttl=63 time=1.141 ms
64 bytes from 192.168.1.100: seq=3 ttl=63 time=1.076 ms
64 bytes from 192.168.1.100: seq=4 ttl=63 time=1.063 ms
64 bytes from 192.168.1.100: seq=5 ttl=63 time=1.123 ms
64 bytes from 192.168.1.100: seq=6 ttl=63 time=1.129 ms
64 bytes from 192.168.1.100: seq=7 ttl=63 time=1.166 ms
64 bytes from 192.168.1.100: seq=8 ttl=63 time=1.153 ms
64 bytes from 192.168.1.100: seq=9 ttl=63 time=1.134 ms
64 bytes from 192.168.1.100: seq=10 ttl=63 time=1.177 ms
64 bytes from 192.168.1.100: seq=11 ttl=63 time=1.077 ms
```

## Registro log del ping de un vistazo

Puesto que utilicé el comando service scale para configurar el estado deseado y ejecutar 5 instancias del servicio ping-service, éste genera nuevos contenedores para constituir las 5 instancias de servicio. Ten en cuenta que swarm realiza el balanceo de carga repartiendo la carga a través de todos los nodos del swarm. Cuando detengo los nodos c2-swarm-03 y c2-swarm-04, espero que swarm rote los servicios en los nodos restantes para mantener el recuento de 5 réplicas. De hecho, este es el caso que podemos ver en la siguiente figura. Swarm está ejecutando 2 contenedores en c2-swarm-00 y c2-swarm-01, y 1 en c2-swarm-02.

Cuando hayas terminado con el servicio, elimínalo introduciendo el siguiente comando:

```
$ docker service rm pingservice
```

```

ayuen@c2-swarm-00 ~
ayuen@c2-swarm-00:~$ docker service scale pingservice=5
pingservice scaled to 5
ayuen@c2-swarm-00:~$ docker service ps pingservice
ID           NAME      IMAGE
ERROR
ayen1j36c88135u4u0f07ixvq pingservice.1 mrdreambot/arm64-busybox-httdp c2-swarm-01 Running
Running 37 seconds
apo pingservice.2 mrdreambot/arm64-busybox-httdp c2-swarm-04 Running
Running less than
a second ago
dqv99sw3s3g2ltqteqrnlip86 pingservice.3 mrdreambot/arm64-busybox-httdp c2-swarm-02 Running
Running 1 seconds
ago
qglgbq76140ehulrbig7n1 pingservice.4 mrdreambot/arm64-busybox-httdp c2-swarm-00 Running
Running 2 seconds
ago
shid5k4y1lfk78pkwiaqdar pingservice.5 mrdreambot/arm64-busybox-httdp c2-swarm-03 Starting
Starting 2 seconds
ago
ayuen@c2-swarm-00:~$ Just shut down c2-swarm-03 and c2-swarm-04
ayuen@c2-swarm-00:~$ docker service ps pingservice
ID           NAME      IMAGE
ERROR
ayen1j36c88135u4u0f07ixvq pingservice.1 mrdreambot/arm64-busybox-httdp c2-swarm-01 Running
Running 5 minutes
tes ago
arm4zwyfulidqmhn9rf5ilm pingservice.2 mrdreambot/arm64-busybox-httdp c2-swarm-01 Running
Running 49 seconds
ago
d7o7ewxc6a592y13bx6dc \_ pingservice.7 mrdreambot/arm64-busybox-httdp c2-swarm-04 Shutdown
Running 4 minutes
tes ago
dqv99sw3s3g2ltqteqrnlip86 pingservice.3 mrdreambot/arm64-busybox-httdp c2-swarm-02 Running
Running 4 minutes
tes ago
qglgbq76140ehulrbig7n1 pingservice.4 mrdreambot/arm64-busybox-httdp c2-swarm-00 Running
Running 4 minutes
tes ago
q292mqlueqiu72f763qkwuu pingservice.5 mrdreambot/arm64-busybox-httdp c2-swarm-00 Running
Running 2 minutes
tes ago
shid5k4y1lfk78pkwiaqdar \_ pingservice.5 mrdreambot/arm64-busybox-httdp c2-swarm-03 Shutdown
Running 4 minutes
tes ago
ayuen@c2-swarm-00:~$ 

```

**Los estados deseados realizando el balanceo de carga**

## Probando la red de enrutamiento

En la Parte 1, pusimos en práctica MySQL y Fish en C2-swarm-00 usando Docker run, en lugar del modo Swarm. En esta sección, voy a hacer uso de ellos como servicios en modo Swarm.

Ten en cuenta que sólo se puede ejecutar al mismo tiempo una única instancia de MySQL asignada al directorio de la base de datos. ¿Recuerdas en la parte 1 que solo c2-swarm-00 tenía un disco duro? Voy a ejecutar MySQL como un servicio con sólo 1 instancia y configurar una restricción de que el servicio sólo se puede ejecutar en el host c2-swarm-00 con el siguiente comando:

```

$ docker service create \
--name mysql \
-p 3306:3306 \
-e MYSQL_USER=fishuser \
-e MYSQL_PASSWORD=fish456 \
-e MYSQL_DATABASE=fish \
--constraint 'node.hostname == c2-swarm-00' \
--mount type=bind,src=/media/sata/fish-mysql,dst=/u01/
my3306/data \
mrdreambot/arm64-mysql

```

Esto es muy similar a ejecutar MySQL usando el clásico comando docker run. La principal diferencia es que si el contenedor que ejecuta el servicio MySQL cae, swarm iniciará automáticamente otra instancia para reemplazarlo.

También tienen que tener en cuenta que usar “--constraint ‘node.hostname == c2-swarm-00’”, que obliga MySQL a ejecutarse solamente en el nodo denominado c2-swarm-00. Existen otras restricciones predefinidas que las puedes encontrar en la documentación de docker. Para ejecutar fish como un servicio, utiliza el comando:

```

$ docker service create \
-p 8080:8080 \
--name fish \
-e MYSQL_SERVER=192.168.1.100 \
-e MYSQL_PORT=3306 \
mrdreambot/arm64-fish

```

Puedes ver que los servicios se ejecutan utilizando el siguiente comando:

```
$ docker service ls
```

Echa un vistazo a lo que aparece cuando tienes éxito. Ten en cuenta que la primera vez que introduces el comando “docker service ls”, MySQL aún se está iniciando. Se inició cuando utilice el segundo comando ‘docker service ls’. Fish se está ejecutando en c2-swarm-04.

Todos los nodos de un swarm están conectados a una red de enrutamiento de entrada, lo que significa que todos los nodos del swarm pueden aceptar conexiones en los puertos publicados para cualquier servicio (en nuestro caso, el puerto 8080 para Fish y el puerto 3306 para MySQL) incluso si la tarea no se está

```

ayuen@c2-swarm-00 ~
ayuen@c2-swarm-00:~$ docker service create \
--name mysql \
-p 3306:3306 \
-e MYSQL_USER=fishuser \
-e MYSQL_PASSWORD=fish456 \
-e MYSQL_DATABASE=fish \
--constraint 'node.hostname == c2-swarm-00' \
--mount type=bind,src=/media/sata/fish-mysql,dst=/u01/my3306/data \
mrdreambot/arm64-mysql
ayuen@c2-swarm-00:~$ docker service create \
-p 8080:8080 \
--name fish \
--MYSQL_SERVER=192.168.1.100 \
--MYSQL_PORT=3306 \
mrdreambot/arm64-fish
ayuen@c2-swarm-00:~$ docker service ls
ID           NAME      REPLICAS IMAGE
mysql.0      1/1      mrdreambot/arm64-fish
e1intxsf4t7r mysql    1/1      mrdreambot/arm64-mysql
ayuen@c2-swarm-00:~$ docker service ps mysql
ID           NAME      REPLICAS IMAGE
e1intxsf4t7r mysql    1/1      mrdreambot/arm64-fish
ayuen@c2-swarm-00:~$ docker service ps fish
ID           NAME      IMAGE
e2l1bhk6brq fish     1/1      mrdreambot/arm64-fish
ayuen@c2-swarm-00:~$ docker service ls
ID           NAME      IMAGE      NODE      DESIRED STATE CURRENT STATE      ERROR
e2l1bhk6brq fish     mrdreambot/arm64-fish c2-swarm-04 Running   Running 3 minutes ago
ayuen@c2-swarm-00:~$ 

```

**Balanceo de carga activo con un Swarm Docker**

ejecutando en ese nodo. Esto implica que podemos apuntar nuestro navegador a cualquiera de los nodos del swarm para acceder a la aplicación fish. Sin embargo, este no fue el comportamiento que observé en mi clúster swarm. Sólo podía acceder al servicio en los nodos que lo ejecutaban (c2-swarm-04) y no en ningún otro nodo. Tras reducir y aumentar el número de réplicas del servicio varias veces, ocasionalmente ni siquiera podía invocar el servicio en el nodo que ejecutaba el servicio. Busqué por Internet y muchas otras personas comentaba que tenían el mismo problema. Se pensaba que el problema estaba en el modo swarm que no actualizaba correctamente las tablas IPVS (servidor virtual IP). IPVS es el módulo del kernel responsable del balanceo de carga. El balanceo automático de carga es una gran función una vez que está funcionando.

## Redes definidas por el usuario

Las redes definidas por el usuario pueden utilizarse para aislar contenedores. Por ejemplo, si creaste una red definida por el usuario y le asignaste contenedores o servicios, otros contenedores o servicios que no pertenezcan a esa red no podrán acceder a ellos y viceversa. Los contenedores o servicios en una red definida por el usuario pueden referirse entre sí por su nombre, como es el nombre asignado a un contenedor al

crearlo. Existen dos tipos de redes definidas por el usuario: redes Bridge y redes Overlay.

## Redes Bridge

Son utilizadas principalmente por los clásicos comandos docker run. Una red bridge sólo puede funcionar en una máquina local. Por ejemplo, podemos crear una red bridge y añadir los contenedores MySQL y Fish:

```
$ docker network create --driver bridge my-net
$ docker run -d --network my-net \
--name mysql \
-e MYSQL_USER=fishuser \
-e MYSQL_PASSWORD=fish456 \
-e MYSQL_DATABASE=fish \
-v /media/sata/fish-mysql:/u01/my3306/data \
mrdrreambot/arm64-mysql
$ docker run -d --network my-net \
-p 8080:8080 \
--name fish \
-e MYSQL_SERVER=mysql \
-e MYSQL_PORT=3306 \
mrdrreambot/arm64-fish
```

Tras crear la red, podrás ver una red bridge (my-net), como la que se muestra en la siguiente Figura.

Ten en cuenta que el comando docker run para Fish hace referencia a MYSQL\_SERVER por su nombre “mysql” en lugar de

```
ayuen@c2-swarm-00:~$ docker network ls
NETWORK ID      NAME    DRIVER      SCOPE
25a9ead77818   bridge   bridge      local
c1deeb45510c   docker_gwbridge   bridge      local
85pv0vkqgeb9   fish-net   overlay    swarm
4f65091046cc   host     host       local
eh587gsaf03u   ingress   overlay    swarm
593c4ae5ac7c   my-net   bridge      local
dc1b1890dfd58   none     null       local
ayuen@c2-swarm-00:~$
```

### Un vistazo a la red bridge

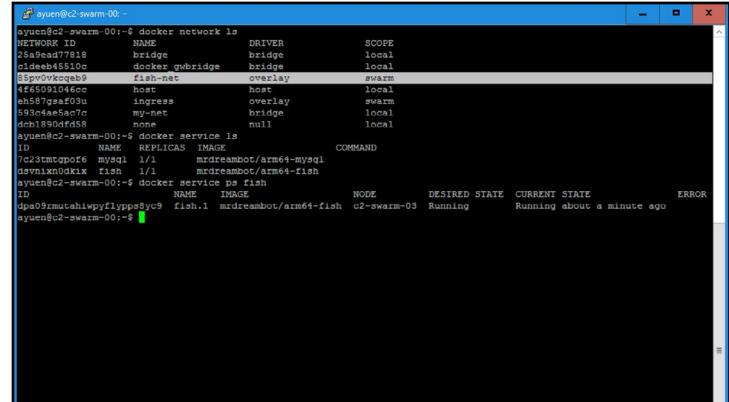
usar localhost o una dirección IP, y ni siquiera tenemos que publicar el puerto MySQL (es decir, no hay que introducir “-p 3306:3306” en el comando MySQL docker run). Aunque no podemos acceder a MySQL desde el host docker sin publicar el puerto 3306, Fish puede acceder al puerto 3306 de MySQL porque está en la misma red definida por el usuario llamada my-net. La red bridge está funcionando según lo previsto.

## Red Overlay

Para crear una red definida por el usuario que abarque múltiples nodos en modo swarm, debemos crear una red Overlay. El modo swarm equivalente a desplegar MySQL y Fish en una red overlay es:

```
$ docker network create --driver overlay --subnet=172.20.0.0/16 fish-net
$ docker service create --network fish-net \
--name mysql \
-e MYSQL_USER=fishuser \
```

```
-e MYSQL_PASSWORD=fish456 \
-e MYSQL_DATABASE=fish \
--constraint 'node.hostname == c2-swarm-00' \
--mount type=bind,src=/media/sata/fish-mysql, \
dst=/u01/my3306/data \
mrdrreambot/arm64-mysql
$ docker service create --network fish-net \
-p 8080:8080 \
--name fish \
-e MYSQL_SERVER=mysql \
-e MYSQL_PORT=3306 \
mrdrreambot/arm64-fish
```



### La configuración de la red overlay

Una vez creada la red, puedes ver una red overlay (fish-net) que ha sido configurada para swarm, como se muestra en la Figura anterior. Desafortunadamente, Fish no funciona. Se muestra la pantalla de inicio de sesión, pero no fue capaz de autenticarse después de introducir el nombre de usuario y contraseña, y no fue capaz de comunicarse con MySQL a través de su nombre. Existe un problema de red overlay en la versión de Docker que utilice, donde la implementación no es capaz de resolver el contenedor o servicio por el nombre.

## Conclusión

Aunque aparecieron algunos problemas con el modo swarm en la versión de docker que usé en este tutorial, debería darte una idea de cómo funciona y lo impresionante que son sus características en un entorno de producción. El modo Swarm integra la gestión nativa del clúster y las funciones de organización de servicios incorporadas en el motor Docker desde la versión 1.12.0. Antes de que apareciera el modo swarm, crear un swarm implicaba usar herramientas de terceros como consul o etcd para proporcionar un almacén de estado distribuido para descubrir el servicio. Con el modo swarm, todo está integrado y funciona correctamente, excepto las cuestiones que hemos comentado anteriormente: la red de enrutamiento y la red overlay definida por el usuario. Aunque este tutorial está diseñado para ejecutarse sobre un swarm ODROID-C2, todos los comandos que hemos utilizado son exactamente los mismos que se usan en las máquinas INTEL que ejecutan el motor Docker. Puedes aplicar fácilmente tus conocimientos de línea de comandos Docker en diferentes entornos como Linux, MacOS, Windows y en un host en la nube.

# DISEÑANDO TU PROPIO SEEDBOX ODROID

## SACALE EL MAXIMO PARTIDO AL CLOUDSHELL

por Joshua Sherman



**L**os Seedbox están lejos de ser algo nuevo, aunque todavía no he encontrado uno que haga todo lo que quiero con mi ODROID-XU4, especialmente cuando lo integro con la carcasa CloudShell y la interfaz LCD. Esta guía te ayudará a montar un seedbox:

### Alimentado por el ODROID-XU4

**Utiliza conectividad SSD con SATA para las transferencias de alta velocidad y las velocidades de lectura/escritura**

**Ejecuta el sistema operativo y el caché en el disco duro para reducir la carga de la tarjeta SD**

**Soporta acceso web a Transmission para realizar descargas en remoto**

**Se conecta automáticamente a una VPN al arrancar para disponer de máxima seguridad**

Aunque nada de esto es original, aquí tienes una forma de éxito garantizado para empezar con un ODROID-XU4 y un kit CloudShell, y terminar con un seedbox con todas estas excelentes características.

### Antecedentes

Para facilitarle las cosas a mi padre, le monte un dispositivo de almacenamiento en red (NAS) el año pasado para cubrir todas sus necesidades de descarga y almacenamiento. Utilice una Raspberry Pi 2 que había encontrado por ahí antes de que descubriera el más potente (y con capacidad USB 3.0) ODROID-XU4. Utilicé ese RPi 2 para desarrollar un seedbox que era un buen comienzo, pero utilice un disco duro USB no muy rápido.

También me di cuenta de lo cómodo y bien empaquetado que está el CloudShell de ODROID, así que decidí juntar dos grandes cosas y escribir una guía sobre cómo convertir Cloudshell en un seedbox con capacidades VPN. Existen otras guías por ahí, a menudo orientadas a la RPi o Virtual Private Server (VPS), pero yo prefiero guías llave en mano que no re-

quieran buscar instrucciones que no aborden los matices del ODROID, Arquitectura ARM o herramientas específicas que quiero usar para lograr un desarrollo perfecto. Tenía la intención de que fuera una actualización de la Pi2 más reciente, más rápida y más elegante.

Tiene su merito PiMylife y MakeUseOf, que tienen algunas guías de Raspberry Pi muy útiles que he adaptado al XU4 y CloudShell. También tomé algunos fragmentos e instrucciones de varios miembros de la comunidad ODROID que me han ayudado a configurar correctamente esta guía para XU4 y CloudShell. En conjunto, todo debería funcionar por menos de 200\$, y ofrecer todo lo que desearias de un seedbox casero o una simple unidad NAS.

### Lista de componentes

#### ODROID XU4

**Kit CloudShell XU4 (con un adaptador USB 3.0 a SA TA y una pantalla TFT LC de 2,2 pulgadas)**

#### Batería RTC

**SSD de 256 GB (o cualquier otro dispositivo de almacenamiento de 2,5 pulgadas)**

**Tarjeta micro SD de 32 GB (u otra unidad de disco de SO)**

**Una suscripción VPN con soporte OpenVPN**



**Los componentes y herramientas que necesitaremos para hacer el trabajo**

## VPN

Esta guía incluye un paso para configurar el cifrado de tráfico a través de una red privada virtual (VPN). Recomiendo utilizar una VPN con un seedbox o NAS que estás utilizando para descargar contenido por seguridad y tranquilidad. Sacrificas tus capacidades de velocidad máxima si tienes una conexión de fibra óptica u otra conexión a Internet de alta velocidad, aunque muchos proveedores de VPN pueden ofrecer velocidades bastante buenas a pesar de dirigir todo el tráfico a través de un servidor de terceros. Además, aún puedes acceder localmente a esos archivos, maximizando la conectividad y el valor de la conectividad USB 3.0 a SATA.

## Local o online?

Esta guía es para configurar el seedbox en local. ¿Puedes hacerlo online? Claro, pero te expones a varios riesgos de seguridad. Esta guía incluye contraseñas de texto sin formato (por simplicidad y por el hecho de realizar una implementación sólo en local) para facilitar la configuración de la conexión VPN en cada inicio. Es posible que esto no lo necesites si quieras una VPN, si tu proveedor de servicios VPN no necesita un nombre de usuario ni una contraseña. También hay una forma más segura de hacer esto, así que tienes la libertad para hacer cualquier sugerencia en el post. He elegido este método, porque está basado en uno anterior que sé que funciona, y cumple con mis necesidades personales para tener un NAS local con un 100% de tráfico cifrado.

## Empecemos

En primer lugar, actualizaremos nuestro ODROID, ya que esta guía utiliza la imagen 16.04 Ubuntu MATE pre-configurada:

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

La actualización tardará un tiempo en completarse. Podríamos usar una imagen más pequeña, pero esta guía está diseñada

### Primer plano del hardware del Seedbox



para ser lo más simple posible. ¡Vete a tomarte un café y vuelve cuando haya terminado! La Figura anterior muestra el hardware desde cerca.

A continuación, vamos a configurar la pantalla del CloudShell de nuestro seedbox para asegurarnos de que funciona. Empecemos con smartmontools:

```
$ sudo apt install smartmontools
```

Luego, debemos seguir las instrucciones de ODROID para configurar el búfer de nuestra pantalla TFT en el CloudShell:

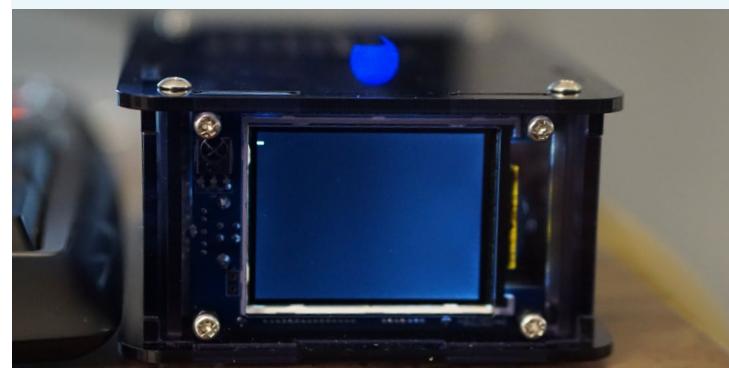
```
$ sudo -s
$ echo "options fbtft_device name=hktft9340 busnum=1
rotate=270" > /etc/modprobe.d/odroid-cloudshell.conf
$ echo "spi_s3c64xx" >> /etc/modules
$ echo "fbftf_device" >> /etc/modules
```

Después, eliminaremos la lista negra de la Interfaz Periférica Serial (SPI) a través de la cual se conectan nuestro ODROID-XU4 y CloudShell. Abriendo blacklist-odroid.conf:

```
$ sudo nano /etc/modprobe.d/blacklist-odroid.conf
```

Cuando hayas terminado, debería verse así. Observa que las dos filas SPI y la fila LCD ahora están comentadas:

```
# Comment the required lines
#
# IO Board
blacklist ioboard_bh1780
blacklist ioboard_bmp180
blacklist ioboard_keyled
#
# SPI
# blacklist spidev
# blacklist spi_s3c64xx
#
# 3.2" LCD Touchscreen driver
blacklist ads7846
```



**¡Tenemos energía! Pero la pantalla aún no funciona**

Reinicia el XU4, pero asegúrate de que no tiene ningún cable HDMI conectado para que se active la pantalla LCD.

Llegados a este punto, tenemos un dispositivo que funciona, pero la pantalla sigue en blanco. Queremos que muestre todo tipo de información útil sobre nuestro CloudShell, así que utilicemos un script para hacer esto. Con el tiempo usaremos /dev/sda2 como unidad de almacenamiento, así que debemos modificar la aplicación de @mdrjr e instalarla:

```
$ sudo apt-get install curl sysstat
$ wget https://github.com/jsherm101/cloudshell_lcd/\
raw/master/cloudshell-lcd_20160913-3-fixed.deb
$ sudo dpkg -i cloudshell-lcd_20160913-3-fixed.deb
```

Si reiniciamos de nuevo, ahora debería mostrar alguna información sobre el dispositivo:

### Uso de la CPU y temperatura

### Disponibilidad de RAM

### La dirección IP local del dispositivo

### Tasas de transferencia

### Uso del disco, que ahora está vacío porque no hemos configurado nuestra unidad /dev/sda2



Las cosas se ven mucho mejor

## Configuración del sistema operativo

A continuación viene la tarea principal de este proyecto, que es configurar una aplicación de protocolo BitTorrent para que funcione correctamente con el SSD. Esto es sólo usar un comando apt-get si queremos usar nuestra tarjeta SD donde está instalado el sistema operativo para el almacenamiento, pero esto presenta dos problemas:

**Existen grandes tarjetas SD (> 256 GB), aunque son más caras o más lentas que sus equivalentes SSD y HDD**

**Una tarjeta SD tiene un ciclo de vida de lectura/escritura mucho más corto que un SSD o HDD, y tiende a fallar antes**

Para solucionar esto, vamos a mover nuestra instalación de ODROID al SSD y arrancar desde el SSD tanto para nuestro sistema operativo como para nuestra carpeta de almacenamiento NAS adicional. Todo el merito se le atribuye a @James por esta gran idea. Puede que no sea una preocupación en la mayoría de los casos, pero si nos despierta un cierto interés en este proyecto dada la gran cantidad de lecturas y escrituras que se preveen en un uso normal.

Para solucionar esto, vamos a mover nuestra instalación de ODROID al SSD y arrancar desde el SSD tanto para nuestro sistema operativo como para nuestra carpeta de almacenamiento NAS adicional. Todo el merito se le atribuye a @James por esta gran idea. Puede que no sea una preocupación en la mayoría de los casos, pero si nos despierta un cierto interés en este proyecto dada la gran cantidad de lecturas y escrituras que se preveen en un uso normal.

A continuación, configuromos nuestra unidad de almacenamiento. Ya la hemos conectado a nuestro CloudShell y los drivers SATA a USB necesarios están configurados con smartmontools, de modo que debería aparecer en fdisk como primera unidad:

```
$ sudo fdisk /dev/sda1
```

Mi unidad es actualmente una unidad NTFS de una instalación antigua de Windows. Presionaremos “d” en fdisk para eliminar la partición, luego “n” para crear una nueva, luego “p” para la partición primaria. Al final crearemos dos particiones: una para el sistema operativo que estamos trasladando al SSD y otra para el área de almacenamiento que compartiremos en la red local.

Crea la primera partición en el primer sector disponible (probablemente 2048) y luego escriba “+ 16G” para crear una partición de 16 GB de tamaño. A continuación, vuelva a escribir “n” y “p” para crear una segunda partición con el resto de almacenamiento de la unidad. Mi SSD de 256 GB dejando unos 208 GB tras tener en cuenta el formato de capacidad y la partición de instalación. Una vez que hayas terminado, escriba “w” para escribir las nuevas particiones en el SSD.

A continuación, convierte ambas particiones en el formato ext4 y móntalas en las dos carpetas que hemos reservado para este proyecto:

```
$ sudo mkfs.ext /dev/sda1
$ sudo mkfs.ext /dev/sda2
$ sudo mkdir /media/systemdrive
$ sudo mkdir /media/NAS1
$ sudo mount /dev/sda1 /media/systemdrive
$ sudo mount /dev/sda2 /media/NAS1
$ df -h (to see a list of drives and confirm everything
is in order)
```

```
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

odroid@odroid:~$ sudo mkdir /media/systemdrive
odroid@odroid:~$ sudo mkdir /media/NAS1
odroid@odroid:~$ sudo mount /dev/sda1 /media/systemdrive
odroid@odroid:~$ sudo mount /dev/sda2 /media/NAS1
odroid@odroid:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            741M    0  741M   0% /dev
tmpfs           200M  7.9M 192M   4% /run
/dev/mmcblk0p2   30G  3.9G  25G  14% /
tmpfs           996M    0  996M   0% /dev/shm
tmpfs            5.0M  4.0K  5.0M   1% /run/lock
tmpfs           996M    0  996M   0% /sys/fs/cgroup
/dev/mmcblk0p1   128M  13M  116M  10% /media/boot
tmpfs           200M  4.0K  200M   1% /run/user/110
tmpfs           200M    0  200M   0% /run/user/1000
/dev/sda1        16G  44M  15G   1% /media/systemdrive
/dev/sda2       219G  60M  208G   1% /media/NAS1
odroid@odroid:~$
```

### Nuestras particiones correctamente configuradas y montadas

Una vez que hayas terminado esto, la configuración de la unidad debería ser similar a la anterior imagen.

Ahora podemos usar rsync para trasladar nuestros datos desde la tarjeta SD al SSD. Antes de hacer esto, debemos hacer una copia de seguridad del archivo boot.ini que terminaremos editando y que está en la partición de arranque FAT de la tarjeta SD:

```
$ sudo cp /media/boot/boot.ini /media/boot/boot.ini.bak
```

Ahora necesitamos encontrar el UUID de la única partición que creamos antes.

```
$ lsblk -f
```

Mi UUID del disco es 7d62ae29-a3cf-41d0-9127-065cf08fbe6, que se utiliza como ejemplo. A continuación, abre boot.ini y busca “Basic Ubuntu Setup”. Puedes buscar en Nano con CTRL + W:

```
$ sudo nano /media/boot/boot.ini
```

Comenta la línea que está justamente por debajo de esa línea. Esta es nuestra configuración inicial para arrancar la tarjeta SD. Siempre podemos revertir el proceso si algo sale mal conectando la tarjeta SD a cualquier dispositivo (PC o \*nix) y volviendo a editar el archivo boot.ini. A continuación, añadiremos nuestras propias instrucciones:

```
## Boot from USB device
setenv bootrootfs "console=tty1 consoleblank=0
root=UUID=7d62ae29-a3cf-41d0-9127-065cf08fbe6 root-
wait rootdelay=10 ro fsck.repair=yes"
```

Simplemente asegúrate de que el UUID sea cualquier UUID para tu partición del sistema de 16 GB en /dev/sda1. Antes de usar rsync para transferir la nueva partición, necesitamos editar

fstab para montar nuestras nuevas unidades al inicio:

```
$ sudo nano /etc/fstab
```

Comenta la primera línea y ahora añade estas dos líneas a continuación, suponiendo que has utilizado los mismos nombres de partición y carpetas que yo:

```
$ /dev/sda1 / ext4 defaults,noatime 0 1
$ /dev/sda2 /media/NAS1 ext4 defaults 0 2
```

Finalmente, podemos trasladar nuestro sistema operativo desde la tarjeta SD a la nueva partición que hemos reservado para ello. Ten en cuenta que una vez hecho esto, cualquier cambio en el sistema operativo no aparecerá tras reiniciar el equipo, ya que estas trabajando desde la nueva partición. Lo mejor es reiniciar una vez que termines con el comando rsync:

```
$ sudo apt-get install rsync
$ sudo rsync -axv / /media/systemdrive
```

Possiblemente tendrás tiempo para tomar café mientras se esté realizando la transferencia. Para ser más específico, estamos copiando todo nuestro sistema operativo en /media/systemdrive, donde hemos montado nuestra partición /dev/sda1. Tenga en cuenta que tras reiniciar, nuestro fstab + boot.ini volverá a montar y redireccionará la partición /dev/sda1 a “/” y actuará como nuestro sistema operativo. Seguiremos usando la partición de arranque de tu tarjeta SD, así que no la

```
odroid@odroid:~$ lsblk -f
NAME      FSTYPE LABEL UUID                                     MOUNTPOINT
sda
└─sda1    ext4   boot 7d62ae29-a3cf-41d0-9127-065cf08fbe6 /
└─sda2    ext4   e7cf4050-dc2d-4747-acb5-d765d67d6e68 /media/NAS1
mmcblk0
└─mmcblk0p1 vfat   boot 52AA-6867                           /media/boot
└─mmcblk0p2 ext4   rootfs e139ce78-9841-40fe-8823-96a304a09859
odroid@odroid:~$
```

### El SSD está montado como sistema de archivos raíz

elimines. Una vez finalizado, es hora de reiniciar y contener la respiración:

```
$ sudo reboot
```

Si se reinicia correctamente, es que definitivamente estamos trabajando desde una de nuestras dos particiones, ya sea en la tarjeta SD o SSD. Puedes confirmar que estamos usando el SSD revisando nuestras particiones de nuevo:

```
$ lsblk -f
```

Debería ver /sda1 montado en “/” para confirmar nuestro éxito. Puedes verlo por ti mismo en la figura anterior.



Our Seedbox is starting to look pretty sharp!

## Configuración de Transmission

De acuerdo, ya tenemos nuestro sistema operativo transferido, y nuestra pantalla muestra nuestro SSD de 256 GB en condiciones de trabajo.

Ahora necesitamos instalar Transmission y configurar nuestro demonio transmission. Usaremos nuestro usuario pre-determinado “odroid”, y configuraremos igualmente nuestro disco para tener una carpeta incompleta y completa. Empecemos instalando Transmission y añadiendo nuestras carpetas incompletas y completas:

```
$ sudo apt-get install transmission-daemon
$ sudo mkdir -p /media/NAS1/incomplete
$ sudo mkdir -p /media/NAS1/complete
```

Ahora necesitamos configurar Transmission desactivando sus servicios y abriendo el archivo de configuración:

```
$ sudo service transmission-daemon stop
$ sudo nano /etc/transmission-daemon/settings.json
```

Tiene que configurar varios parámetros:

**Configurar como incompleto “true” y “media/NAS1/incomplete”**

**Configurar como completo “media/NAS1/complete”**

**Ajustar rpc-authentication-required a “false”**

**Configurar la lista blanca en 192.168. \*. \* para asegurarte de que puedes acceder a ella remotamente desde otro dispositivo de tu red**

Cada una de las filas que hemos editado debe tener este aspecto, distribuidas a lo largo del archivo de configuración:

```
"download-dir": "/media/NAS1/complete",
```

```
"incomplete-dir": "/media/NAS1/incomplete",
"incomplete-dir-enabled": "true",
"rpc-authentication-required": "false",
"rpc-whitelist": "127.0.0.1,192.168.*.*,10.0.*.*",
```

También hay otros parámetros que puede ajustar, te recomiendo que consultes el sitio web de Transmission para obtener más información sobre cómo configurar esta herramienta.

Antes de iniciar nuestro demonio transmission, necesitamos cambiar la propiedad de nuestro usuario “odroid” para que todo funcione correctamente. Existen mejores formas de hacer esto, pero prefiero continuar con un solo usuario ya que este servidor no tendrá ningún otro propósito que no sea un seedbox y no veo la necesidad de usar el usuario del demonio transmission que normalmente se suele utilizar.

```
$ sudo chown -R odroid:odroid /etc/transmission-daemon
$ sudo chown -R odroid:odroid /etc/init.d/transmission-daemon
$ sudo chown -R odroid:odroid /var/lib/transmission-daemon
$ sudo chown -R odroid:odroid /media/NAS1/
```

También necesitamos abrir el servicio demonio de transmission y ajustar el user = “odroid” desde “transmission-daemon”:

```
$ sudo nano /etc/systemd/system/multi-user.target.wants/transmission-daemon.service
```

También queremos hacer esto en el archivo init.d, cambiando “odroid” por “USER”:

```
$ sudo nano /etc/init.d/transmission-daemon
```

Finalmente, reinicia el demonio y vuelve a activar Transmission:

```
$ sudo systemctl daemon-reload
$ sudo service transmission-daemon start
```

## Configuración de Samba

Ahora tenemos todo lo que necesitamos para descargar un archivo, pero nos hace falta una forma de acceder a los archivos que descargamos a través de nuestra red local. Si estás utilizando un dispositivo Windows, Samba es la clave. Empecemos por instalarlo y configurar nuestro recurso compartido en red. Necesitaremos configurarlo de una forma que nos permita iniciar sesión con nuestro usuario odroid y descargar los archivos con facilidad, así como eliminar los archivos que ya no queremos en nuestro seedbox:

```
$ sudo apt-get install samba samba-common-bin
$ sudo nano /etc/samba/smb.conf
```

Añade lo siguiente como una nueva configuración para compartir nuestras carpetas importantes:

```
security = user

[odroid]
comment = odroid
path = /media/NAS1
valid users = @odroid
force group = odroid
create mask = 0775
force create mode = 0775
security mask = 0775
force security mode = 0775
directory mask = 2775
force directory mode = 2775
directory security mask = 2775
force directory security mode = 2775
browseable = yes
writeable = yes
guest ok = no
read only = no
```

Antes de seguir adelante, tendremos que configurar una contraseña de Samba para nuestro usuario “odroid”. También puedes crear cuentas independientes si lo deseas, siempre y cuando esos usuarios estén en el grupo “odroid”:

```
$ sudo smbpasswd -a odroid
```

Ahora podemos reiniciar, y deberíamos poder acceder remotamente a nuestras carpetas de seedbox.

```
$ sudo service smbd restart
```

Ten en cuenta que siempre puedes añadir nuevos usuarios compatibles, siempre y cuando configures una contraseña adecuada y añadas esos usuarios al grupo “odroid” o a cualquier grupo al que le asignes acceso a estos archivos.

## Configuración de VPN

Por último, queremos configurar nuestra VPN para que automáticamente se conecte de forma segura al arrancar. Una vez más, gracias a James @ MakeUseOf que descubrió una forma rápida y fácil de conseguir una VPN funcional sin necesidad de iniciar manualmente la VPN o introducir las credenciales cada vez que reinicies el dispositivo. Esto es esencial si deseas garantizar el cifrado de tu conexión mientras utilizas el seedbox.

En esta guía, estamos utilizando una conexión OpenVPN, que depende de:

**Un archivo de configuración de OpenVPN**

**Un certificado de tu proveedor VPN**

**Tu nombre de usuario y contraseña para tu proveedor de VPN almacenados en un archivo de texto**

**Tres scripts shell especiales para iniciar nuestra VPN automáticamente al arrancar y enrutar el tráfico.**

Almacenar una contraseña en texto no es lo más idóneo. Sin embargo, es la manera más rápida y más sencilla de empezar a funcionar y puesto que sólo estamos en local, tener las credenciales de VPN almacenadas de esta forma no tiene riesgo para los archivos a los que accedemos desde una red doméstica. Si deseas alojar tu seedbox con redirección de puertos y que esté básicamente abierto a Internet, te recomiendo que busque un método alternativo o tener en cuenta estos riesgos antes de continuar.

A continuación, vamos a instalar el software OpenVPN:

```
$ sudo apt-get install openvpn resolvconf
```

Vamos a dejarlo todo en nuestra carpeta de inicio, que es /home/odroid en este caso. Realmente puede estar en cualquier lugar, pero estos no son archivos que quieras que estén en el mismo lugar desde donde la gente de tu red doméstica se conectaría y descargaría los archivos de tu seedbox.

En primer lugar, consigue tu archivo de configuración de OpenVPN (denominado vpn-server.ovpn en esta guía) y colócalo en tu carpeta de inicio junto con tu archivo de certificado (denominado ca.crt en esta guía). Ahora crea un nuevo archivo de texto (.txt) con dos líneas. La primera línea debe ser el nombre de usuario de tu servicio de conexión VPN y la segunda línea debe ser la contraseña del servicio de conexión VPN:

```
$ sudo nano /home/odroid/pass.txt
$ username
$ password
```

A continuación, vamos a abrir nuestro archivo de configuración, que en mi caso es “vpn-server.ovpn”:

```
$ sudo nano /home/odroid/vpn-server.ovpn
```

Vamos a agregar la siguiente línea en la parte inferior, que nos permitirá conectarnos al proveedor VPN sin introducir las credenciales manualmente:

```
$ auth-user-pass /mnt/torrents/openvpn/pass.txt
```

A continuación, vamos a añadir estas tres líneas que hacen referencia a los scripts shell que estamos a punto de crear:

```
$ route-up /home/odroid/route-up.sh
$ down-pre
$ down /home/odroid/down.sh
```

Esto permitirá que OpenVPN se conecte automáticamente a nuestro proveedor VPN. Una vez que se conecte, automáticamente enrutaría todo el tráfico a través de esta conexión VPN, ya sea entrante (down.sh) o saliente (route-up.sh). A continuación, vamos a crear estos dos scripts shell:

```
$ sudo nano /home/odroid/route-up.sh
```

Escriba lo siguiente para route-up.sh:

```
#!/bin/sh
iptables -t nat -I POSTROUTING -o tun0 -j MASQUERADE
```

Después, edita down.sh:

```
$ sudo nano /home/odroid/down.sh
```

Escriba lo siguiente para down.sh:

```
#!/bin/sh
$ iptables -t nat -D POSTROUTING -o tun0 -j MASQUERADE
```

Por último, necesitamos un script shell que podamos lanzar cuando nuestro sistema operativo arranque y se utilice para iniciar la conexión VPN automáticamente:

```
$ sudo nano /home/odroid/vpn.sh
```

Escriba lo siguiente para vpn.sh:

```
$ sudo openvpn --client --config /home/odroid/vpn-
server.ovpn --ca /home/odroid/ca.crt --script-security
2
```

Ahora, vamos a hacer todos estos archivos ejecutables:

```
$ sudo chmod +x /home/odroid/route-up.sh
$ sudo chmod +x /home/odroid/down.sh
$ sudo chmod +x /home/odroid/vpn.sh
```

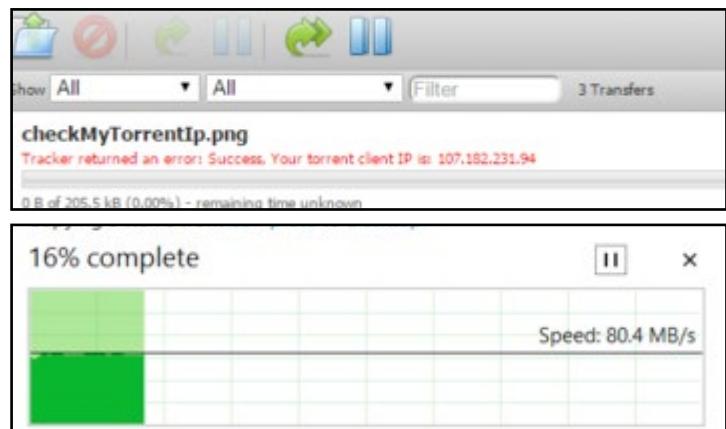
Ahora vamos a abrir nuestro archivo rc.local, que ejecuta los scripts al iniciar el sistema:

```
$ sudo nano /etc/rc.local
[code]
```

Agrega esta línea justamente encima de la línea “exit 0”:

```
[code]
$ /home/odroid/vpn.sh
```

Tu VPN ya está configurada y tu seedbox descargará exclusivamente a través de una conexión cifrada con tu proveedor



#### El entorno de transmisión: ¡Observa estas velocidades!

de VPN. Puedes hacer pruebas fácilmente usando el archivo de prueba TorGuard que verifica tu dirección IP al descargar el archivo.

Conéctate a tu interfaz web de Transmisión en odroid: 9091, o cualquiera que sea tu dirección IP local.

Carga el archivo de prueba TorGuard y comprueba la dirección IP reportada como “Success! Your torrent client IP is XX.XX.XX.XX” Verifica esta dirección IP en comparación con la que Google informa como tu dirección IP visitando www.whatismyip.com.

Si las direcciones IP son diferentes, entonces es que tu conexión es segura. Todavía puedes usar tu dirección IP local para acceder y descargar archivos a través de Samba. Mi proveedor de VPN me permite descargar a velocidades de 75 Mbps, con el ODROID-XU4 alcanzando un 22% de uso de la CPU, lo que demuestra lo potente que es el procesador octa-core para tareas como éstas. Sin embargo, lo mejor es la velocidad de transferencia de tu ODROID a tu ordenador personal. Puede observar velocidades de transferencia de hasta 800 Mbps al mover algunos archivos de ida y vuelta, haciendo que seedbox sea un dispositivo de almacenamiento en red extremadamente viable.

Tomate la libertad de hacer cualquier comentario o sugerencia en el tema del foro en <http://bit.ly/2fPeek7>. La configuración no es perfecta, pero es infalible para que cualquiera pueda empezar con su propio seedbox.

# CONOCIENDO UN ODROIDIAN

## DANIEL HAZE (@FOURDEE)

editor por Rob Roy (@robroy)

*Por favor, háblanos un poco sobre ti.*

Vivo en Burton Latimer en Inglaterra. Es uno de esos pueblos “yeolde”, un lugar encantador para vivir y educar a una familia. Actualmente soy un “papá hogareño”, ya que estoy haciendo frente a una enfermedad gastrointestinal a largo plazo que afecta y limita mi vida diaria. Antes de eso, era un auxiliar TI para una compañía de drenaje cerca de Cambridge. Tengo un hijo de dos años y medio llamado Jack, al que le encanta el programa de televisión “PAW Patrol”, jugar con los cubos de basura (todavía ni idea de por qué) y construir cosas con los Legos. Gemma, mi novia, es una madre a tiempo completo. Mi formación académica está centrada en las Tecnologías de la Información y las Comunicaciones. La escuela no era realmente para mí, ya que soy más bien una persona “práctica” que disfruta desmontando cosas para ver de lo que son capaces de hacer.

*¿Cómo empezaste con los ordenadores?*

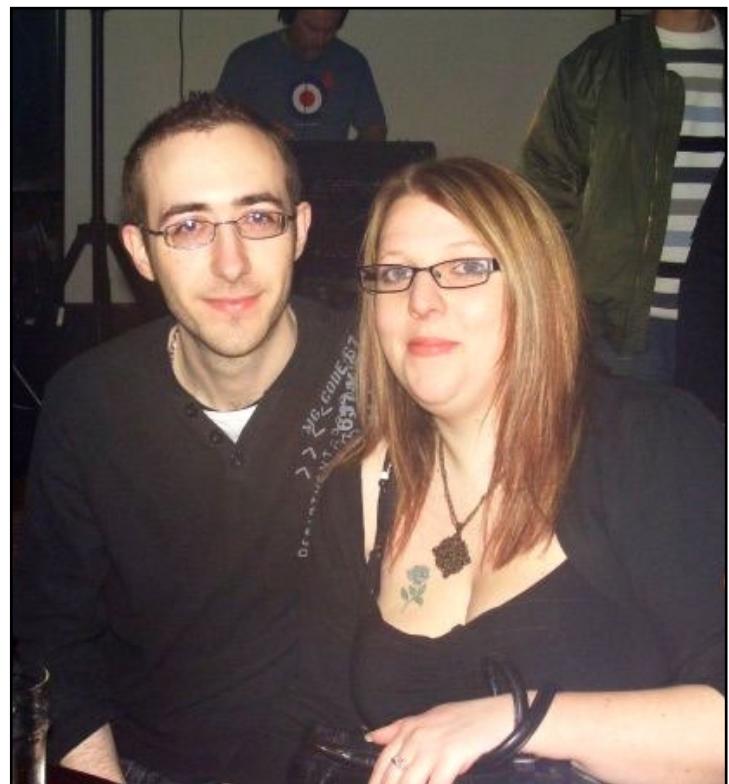
Mi primera experiencia con los ordenadores fue con el Commodore 64 cuando tenía alrededor de 5 o 6 años. Sin embargo, por alguna razón, los únicos recuerdos que tengo son los de la unidad de cinta que tardaba en cargar más de 2 horas. Aún entonces, tenía suerte si el juego realmente llegaba a cargar. ¡Esos sí que fueron días! También tuve la suerte de tener una Amiga 500+ en mis primeros años. Para mí, este fue el inicio de mi pasión por los ordenadores. Utilicé el rastreador MOD para reproducir la canción Axel F, y recuerdo Workbench, Elite Frontier y no tener la actualización de 1MB RAM para jugar “Rise of the Robots”.

*¿Qué te atrajo de la plataforma ODROID?*

He leído los análisis del C1 y las pruebas de rendimiento comparándolo con la Raspberry Pi 2. Me agarró al instante, y simplemente la machaco por mucho en cada una de las áreas de rendimiento. Desde entonces, me he enganchado a los ODROIDS. Es genial ver que Hardkernel innova continuamente y empuja sus productos a nuevos niveles. Incluso iría aún más lejos diciendo que los ODROIDS están años por delante de cualquier cosa que ofrece actualmente la Fundación Raspberry Pi.

*¿Cómo usas tus ODROIDS?*

Tengo un ODROID-C2 que utilizo para las pruebas diarias con mi imagen de DietPi. Esto se debe principalmente al excelente rendimiento de E/S del C2 con el eMMC, lo cual permite un entorno de pruebas extremadamente rápido, especialmente cuando se trata de pruebas de instalación automatizada



Daniel y su esposa Gemma

de software. También tengo otro C2 con el complemento HiFi Shield 2, que utilizo como sistema de música dedicado con Music Player Daemon y la interfaz YMPD. También tengo un XU4 en una carcasa Cloudshell, que utilizo principalmente para mostrar estadísticas con DietPi-Cloudshell, como almacenamiento red (NAS), copia de seguridad con DietPi-Sync, que está basado en rsync y ejecuta PiHole para bloquear anuncios en nuestra red local.

*¿Cuál es tu ODROID favorito y por qué?*

Sin lugar a dudas, es el ODROID-C2. Para mí, es el mejor ordenador de placa reducida (SBC) por excelencia en el mercado actual. Junto con una conexión Ethernet de 1GB y un eMMC capaz de alcanzar velocidades de transferencia de 140Mb/s, es un monstruo en rendimiento capaz de cualquier cosa, desde un servidor NextCloud a un sistema HiFi. Los ODROIDS han recorrido un largo camino desde el C1 original. Todos esos pequeños problemas con fuentes de alimentación inestables y la incompatibilidad con tarjetas SD se han solucionado. Para mí el C2 es el SBC perfecto.



**La ciudad natal de Burton Latimer de Daniel en Inglaterra**

Tu imagen DietPi es muy popular. ¿Cuál fue lo que te motivo a desarrollarla, y qué mejoras piensas incluirle en el futuro?

DietPi empezó siendo originalmente una imagen mínima. Estaba dirigida a reducir la carga de recursos de las grandes imágenes por defecto proporcionadas por el primer modelo de Raspberry Pi. Teníamos estos dispositivos de 700mhz usando mucho más recursos de lo que realmente necesitaba el usuario. Ahora DietPi ha evolucionado bastante, ofreciendo instalaciones automatizadas de software conocido, todo optimizado y configurado para ti. Con DietPi, consigues el máximo rendimiento de tu dispositivo, ya que sólo estás instalado el software que realmente necesitas..

En cuanto al futuro, DietPi siempre ha sido diseñada atendiendo a los comentarios, sugerencias y soporte de los usuarios finales. Tenemos planes para incluir una interfaz web en DietPi, con el fin de reemplazar el sistema de menú whiptail, y proporcionar una experiencia GUI más moderna para los usuarios.

*¿Qué innovaciones te gustaría ver en futuros productos de Hardkernel?*

Personalmente, me encantaría ver un bus y puerto USB 3.0 en la próxima placa de la serie C. Memoria DDR4 sería sublime y aumentar el rendimiento en general, especialmente con la memoria compartida GPU. WiFi y Bluetooth integrado con eficiencia energética también sería una aportación muy acogida en futuras placas.

*¿Qué aficiones e intereses tiene aparte de los ordenadores?*

La música siempre ha estado cerca de mi corazón. Soy un aficionado a la música con pasión por todos los géneros, aunque especialmente por el trance. Me gusta pasar tiempo con mi familia y amigos. Mi hijo está con los Legos, y es agradable construir cosas juntos. Aunque, incluso cuando está dormido, ¡es fácil dejarse llevar e intentar construir el Halcón Milenario!

*¿Qué consejo le darías a alguien que quiere aprender más sobre programación?*

Si eres nuevo en esto de la programación, no empieces con un montón de código extraño en pantalla. La programación no es tan difícil como podrías pensar. Sigue una guía online para "mi primer programa" y seguir con ella. Con la práctica y paciencia, poco a poco todo ese código extraño tendrá sentido y se convertirá en una pasión.

Para los principiantes, recomiendo Python. Es fácil de aprender, tiene una gran comunidad y muchas guías online para empezar. Además, es el lenguaje de programación líder para proyectos GPIO en los SBCs.

Por último y más importante, independientemente de tu nivel o lenguaje elegido, siempre asegúrate de programar algo que te haga ilusión, y disfrutar de cada momento con ello.



**El hijo fotogénico de Daniel y maestro constructor de Lego, Jack**